uc3m | Universidad Carlos III de Madrid

Grado Universitario en Ingeniería en Tecnologías de Telecomunicación

2018-2019

Trabajo Fin de Grado

"Desarrollo de un sistema de autenticación para control de acceso mediante tecnología Bluetooth"

Lara Ceballos Magán

Director: Ignacio Soto Campos

Codirector: Carlos E. Tejedor Izquierdo

Leganés, febrero de 2019

[Incluir en el caso del interés de su publicación en el archivo abierto]

RESUMEN

Este proyecto consiste en el diseño y desarrollo de un sistema de control de acceso mediante la tecnología Bluetooth Low Energy, con el fin de realizar un prototipo para evaluar su implementación en la empresa Redsys.

El sistema está formado por una aplicación móvil desarrollada con tecnología Android y un dispositivo de control, que a su vez está compuesto por la placa de desarrollo ESP32 DevKitC V4, un LED tricolor y un zumbador. La aplicación Android realiza la función de "autenticador" frente a dispositivo de control de acceso, que se encarga de verificar esa autenticación y autorizar el paso.

En este documento se introducirán las tecnologías y protocolos que se van a emplear para el sistema de control de acceso. A continuación, se explicará detalladamente cómo se han desarrollado los programas implantados en los dos dispositivos principales, cómo se realiza el proceso de autenticación basado en un protocolo de autenticación estándar y, lo más importante, cómo se ha implementado toda la conexión e intercambio de mensajes a través de la tecnología Bluetooth Low Energy. Además, se demostrará el correcto funcionamiento del sistema completo mediante las pruebas realizadas y se introducirán nuevas líneas de trabajado para seguir mejorando el sistema en el futuro.

Palabras clave: Bluetooth Low Energy, Android, control de acceso, protocolo EAP-PSK, seguridad.

ABSTRACT

This project consists of the design and development of an access control system using Bluetooth Low Energy technology, in order to make a prototype to evaluate its implementation in the company Redsys.

The system is composed of a mobile application developed with Android technology and a control device, which is also composed of the development board ESP32 DevKitC V4, a tricolor LED and a buzzer. The Android application performs the function of authenticator against the access control device, which is responsible for verifying that authentication and authorizing the passage.

This document will introduce the technologies and protocols that will be used for the access control system. Next, it will be explained in detail how the programs implemented in the two main devices have been developed, how the authentication process is carried out based on a standard authentication protocol and, most importantly, how the entire connection and exchange of messages has been implemented through Bluetooth Low Energy technology. In addition, the correct functioning of the complete system will be demonstrated through the tests carried out and new lines of work will be introduced to continue improving the system in the future.

AGRADECIMIENTOS

En primer lugar, me gustaría dar las gracias a mis dos tutores, Ignacio Soto y Carlos Tejedor, por el gran apoyo que me han dado durante estos meses. Sin ellos este proyecto no hubiera salido adelante de la misma manera que lo ha hecho.

Además, quisiera agradecer a la empresa Redsys haberme ofrecido la oportunidad de realizar este prototipo allí.

Por último, me gustaría dar las gracias a mis padres por estar siempre ahí y ayudarme a compaginar la universidad con el trabajo, y también a Shane, por apoyarme y animarme siempre que estaba estresada aunque estemos a miles de kilómetros de distancia.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1. Motivación del Trabajo

En este trabajo se ha desarrollado un sistema de autenticación para control de acceso físico a través de una aplicación móvil y empleando tecnología Bluetooth. La motivación de este trabajo surge a partir de la necesidad de actualizar la tecnología de control de acceso de la empresa que ha colaborado en este TFG, Redsys.

Actualmente, la autenticación se lleva a cabo mediante tarjetas inteligentes que se conectan por "contactless" al control de acceso. Esto obliga, tanto a trabajadores como al personal externo, a llevar siempre esa tarjeta encima. Sin embargo, hoy en día lo único que se lleva encima es el teléfono móvil, por lo que parece conveniente proponer el uso de este en la autenticación. Por otra parte, las tarjetas inteligentes tienen la desventaja de que pueden dejar de funcionar por problemas de interferencias y es muy difícil poder recuperarlas, mientras que la reconexión en teléfonos móviles es sencilla y no requiere mucho tiempo ni trabajo.

A partir de esta idea, se valoró usar tecnología NFC, pero debido a que no todas las marcas de móviles la tienen integrada, se descartó y se decidió usar la tecnología Bluetooth, que ofrece características similares en cuanto a la conexión inalámbrica, pero tiene la capacidad de intercambiar un mayor número de datos en cada envío.

Debido a que la tecnología Bluetooth no es segura y se pretende usar en un entorno en el que la seguridad es fundamental, surge la necesidad de implementar un protocolo de comunicaciones seguras que evite ataques de suplantación de identidad y robos de información.

1.2. Objetivos

El principal objetivo del proyecto es diseñar e implementar un sistema de control de acceso basado en la conexión Bluetooth entre un microprocesador, que hace de prototipo de control de acceso, y un Smartphone con sistema operativo Android.

Este objetivo principal se divide en varios sub-objetivos que ayudan a comprender mejor el proyecto que se ha desarrollado:

- Desarrollo de un prototipo de control de acceso utilizando un microprocesador que se autentique mediante tecnología Bluetooth.

- Diseño e implementación de una aplicación móvil para Android que intervenga en el proceso de autenticación y haga el papel de "autenticador" frente al control de acceso.
- Definición e implementación del protocolo seguro de comunicaciones que se utilizará en la conexión entre el control de acceso y el teléfono móvil.

1.3. Estructura de Trabajo

Este documento se presenta en tres grandes partes, que pretenden ofrecer una visión global del problema expuesto. Esta división ayudará a comprender el proceso por el que ha pasado el proyecto:

- Capítulo 1: trata de dar una visión general del proyecto, explicando la motivación que ha llevado a abarcar este asunto y los objetivos que se buscan con él.
- Capítulo 2: detalla la elección de las tecnologías usadas y el porqué de ellas. Además de la elección del protocolo de seguridad a seguir.
- Capítulos 3 y 4: diseño y desarrollo del sistema de control de acceso.
- Capítulo 5: pruebas realizadas para asegurar el correcto funcionamiento de la solución.
- Capítulo 6: explica el impacto socioeconómico que supone el producto desarrollado, el marco regulador que aplica, la planificación que se ha llevado a cabo y el presupuesto del proyecto.
- Capítulo 7: expone las conclusiones a las que se ha llegado con su puesta en funcionamiento y las líneas futuras de mejora.

Además, se añaden dos anexos, el primero con datos relativos a una de las pruebas realizadas y el segundo, que contiene el resumen extendido en inglés.

2. ESTADO DEL ARTE

2.1. Control de acceso y presencia

Un control de acceso es un sistema de verificación que permite o rechaza la entrada de personas, vehículos, sistemas automatizados, etc. a un recurso, tanto físico (edificio, ordenador) como lógico (aplicación móvil, sistema operativo). Particularizando al tema que se expone en este Trabajo, el control de acceso es un servicio que regula el paso de personas a los edificios, plantas y/o departamentos de una empresa.

Por otro lado, un control de presencia se encarga de gestionar, principalmente, las horas trabajadas de los empleados y normalmente, se implementa en el mismo sistema que el control de acceso ya que depende de este.

Debido a la gran demanda de esta herramienta por parte de las empresas y a la velocidad con la que avanza la tecnología, existen muchos tipos: tarjetas inteligentes, controles biométricos, controles por conexión NFC, controles por conexión Bluetooth, etc.

Dentro de los controles por conexión Bluetooth, el sistema de autenticación está compuesto por un "Smartphone" y un dispositivo de control. El teléfono tiene instalada una aplicación móvil que permite la conexión con el dispositivo y la autenticación a través del intercambio de un secreto compartido.

Existen diversas ofertas comerciales basados en Bluetooth que siguen la línea de los expuesto anteriormente, como por ejemplo la implementación de la empresa Kisi [1] y la creada por la empresa Nedap, llamada Mace [2]. Estos sistemas son propietarios y cerrados, por lo que no ha sido posible encontrar información técnica de cómo realizan sus operaciones. Por estos motivos, estos sistemas no servían para Redsys, y era necesario desarrollar un sistema propio.

En la figura 1 de la página siguiente se expone un esquema general de la conexión entre el teléfono móvil y el control de acceso, que es el que siguen la mayoría de las ofertas comerciales.

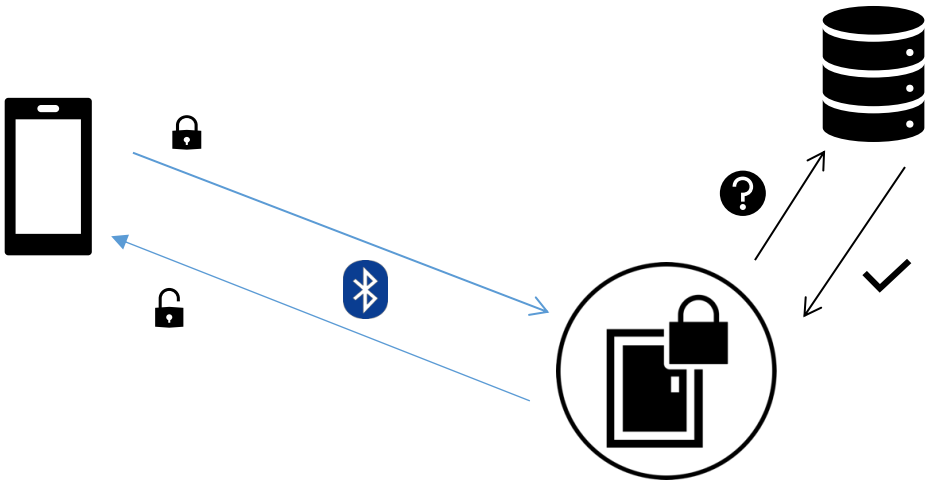

Fig. 1. Esquema de conexión por Bluetooth de un control de acceso

2.1.1. Microprocesador ESP32

Para este proyecto, se han barajado varias alternativas de dispositivos de control de acceso. En un principio, se pensó en utilizar un control de acceso real, pero estos ya vienen con su propia implementación del sistema de autenticación, por lo que no se adecuaba a las necesidades del TFG.

Por otro lado, también se han tenido en cuenta Arduinos y Raspberries, que están destinados a proyectos sencillos de ámbito educativo, que fomentan el aprendizaje de las nuevas tecnologías. Sin embargo, este proyecto tiene un fin profesional, por lo que la tecnología de estos dispositivos no cubre todos los aspectos necesarios.

Es por ello que se ha decidido emplear como prototipo una placa de desarrollo, concretamente la ESP32 DevKitC V4. Se ha elegido esta placa debido a las siguientes características [3]:

- Integración de WiFi y Bluetooth, tanto Classic como Low Energy, en el módulo, por lo que puede funcionar como un sistema independiente completo y reduce la sobrecarga de la pila de comunicaciones en el procesador principal.
- Incluye un puerto USB que proporciona una fuente de alimentación de la placa y una interfaz de programación en serie.
- Incluye un SDK (Software Development Kit) con código fuente específico para la placa y ejemplos de cómo utilizarlo. Además, trabaja en lenguaje C, lo que facilita el desarrollo de aplicaciones.

- Posibilidad de implementar criptografía, incluye un apartado de seguridad con cifrado flash y funciones criptográficas por hardware (SHA, AES, ECC).
- El módulo admite una velocidad de datos de hasta 150 Mbps y una potencia de salida de 20 dBm en la antena, esto es fundamental para el proyecto ya que la autenticación entre el control de acceso y el teléfono móvil tiene que ser rápida.

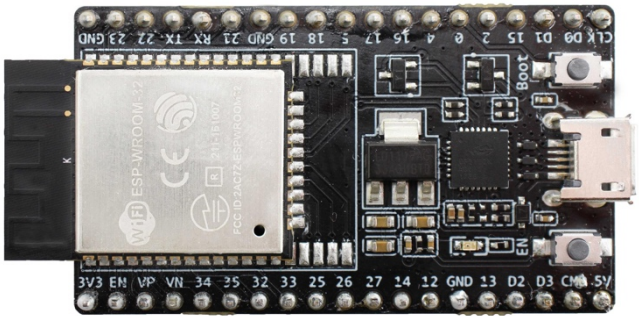

Fig. 2. ESP32 DevKitC V4 board (tomada de [17])

2.2. Bluetooth

Bluetooth es un protocolo de comunicaciones inalámbricas de corto alcance creado en 1994 por Jaap Haartesen y Mattisson Sven, y publicado a través de Bluetooth Special Interest Group (BSIG), con el propósito de emplearse en dispositivos de bajo consumo. Trabaja en la banda de radio frecuencia ISM (Industrial, Scientific and Medical), entre las bandas 2402-2480 MHz y 2400-2483.5MHz.

El protocolo está basado en el envío de paquetes y sigue una estructura cliente/servidor, en la que clientes y servidores se pueden intercambiar los papeles. Esto es importante ya que, de este modo, ambos tienen la posibilidad de iniciar la comunicación; típicamente las comunicaciones solo las pueden empezar los clientes.

Por otra parte, Bluetooth cuenta con varias especificaciones que van modernizando el estándar original. Actualmente, todos los dispositivos, tanto teléfonos móviles como placas de desarrollo, tienen integradas versiones a partir de la 4.0, por lo que son las únicas que se expondrán aquí.

En la versión 4.0 se incluyen los protocolos *Classic Bluetooth*, que sigue la línea de las versiones anteriores de Bluetooth; *Bluetooth High Speed*, basado en Wi-Fi; y *Bluetooth Low Energy* (BLE), que proporciona el mismo rango de cobertura que los anteriores, pero reduciendo considerablemente el consumo de batería.

Las mayores diferencias entre *Classic Bluetooth* y *Bluetooth Low Energy* se plasman en la tabla 1:

TABLA 1. DIFERENCIAS ENTRE CLASSIC BLUETOOTH Y BLE [4]

	BLE	**Classic Bluetooth**
Transmisión de datos	Dividida en pequeñas ráfagas	Continua
Tiempo de conexión	<6ms	100ms
Máximo nº de dispositivos conectados	Ilimitado	7
Velocidad en el envío de datos	125Kb/s a 2Mb/s	1Mb/s a 3Mb/s
Tamaño máximo de carga útil	251 bytes	1021 bytes
Seguridad	128-bit AES, definido por el usuario en la capa de aplicaciones	64b/128b, definido por el usuario en la capa de aplicaciones
Definición de servicio	Perfiles GATT (se explicarán más adelante)	Perfiles tradicionales

Cabe destacar que los datos de la tabla 1 se refieren de a comunicaciones "punto a punto".

Por último, en los controles de acceso se necesita que la comunicación sea rápida ya que en la hora de comienzo y final de las jornadas laborales se acumulan muchas personas. Por ello, estos sistemas utilizan la tecnología BLE, que tiene un tiempo de conexión mucho menor que el Bluetooth tradicional y mejora el intercambio de mensajes debido a la división de estos en pequeños paquetes.

2.2.1. Bluetooth Low Energy

En el apartado superior, se han descrito las características técnicas del Bluetooth Low Energy, por lo que esta sección se centrará en describir su arquitectura y cómo se realiza la conexión entre dos dispositivos. Esta sección está basada en la referencia [5].

2.2.1.1. Arquitectura

En primer lugar, como en el *Classic Bluetooth*, la pila de protocolos tiene dos partes principales: el controlador (Controller) y el anfitrión (Host). A su vez, el *Controller* se divide en la capa física (PHY) y la capa de enlace (LL), y el *Host* en los protocolos y perfiles siguientes:

- L2CAP (Logical Link Control Protocol): da apoyo a la multiplexación, segmentación y reensamblado de paquetes en protocolos de niveles superiores.
- GAP (Generic Access Profile): define los procedimientos genéricos para el descubrimiento de dispositivos y la conexión entre ellos, además de un formato común de parámetro para la interfaz del usuario.
- ATT (Attribute Protocol): define el protocolo para el descubrimiento, lectura y escritura de atributos en un dispositivo.
- GATT (Generic Attibutes): describe una estructura de datos jerárquica construida sobre el protocolo ATT.
- SM (Security Manager): define el protocolo y comportamiento del emparejado, autenticación y encriptación entre dispositivos.

En los apartados siguientes, se describirá en profundidad los protocolos GAP, ATT y GATT.

GAP

El principal objetivo de este perfil genérico es el definir un conjunto de requisitos y recomendaciones relacionados con los modos de acceso que se usará en las demás capas y protocolos. Describe cómo se tienen que comportar los dispositivos, principalmente, en los estados de espera, descubrimiento y conexión para garantizar que los enlaces entre dispositivos se establecen de manera correcta.

En la figura 3 se observa cómo este perfil abarca a todas las capas inferiores de la arquitectura, debido a que es la base de todos los demás perfiles.

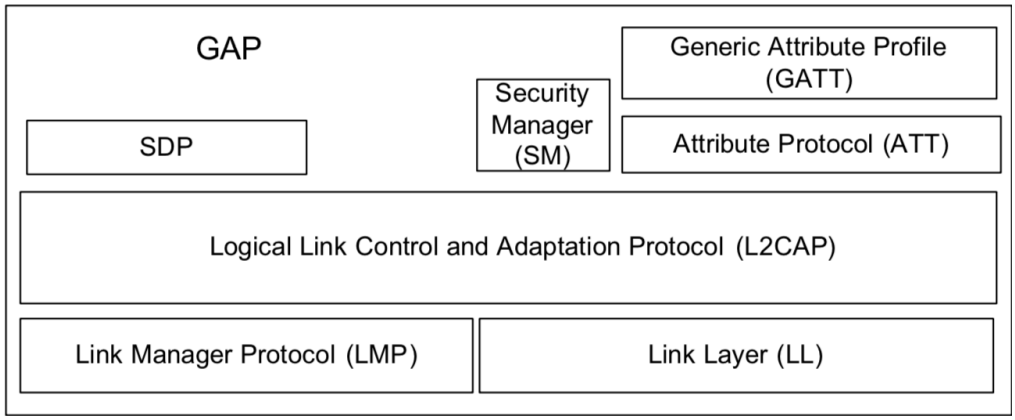

Fig. 3. Relación de GAP con las capas inferiores (tomada de [5])

ATT

Este protocolo permite a un dispositivo, cuyo rol será el de servidor, mostrar una serie de atributos, con sus valores, a otro dispositivo, cuyo rol será el de cliente, para que este los identifique, lea o escriba en ellos.

Un atributo está formado por las siguientes propiedades:

- Tipo de atributo: especifica lo que el atributo representa y está definido por un UUID (identificador único universal o *universally unique identifier*).
- Identificador de atributo: identifica únicamente a un atributo en el servidor, permitiendo que el cliente solicite la lectura o escritura del atributo a través de este parámetro.
- Permisos de atributo: serie de permisos que son gestionados por el GATT: permisos de acceso, de encriptación, de autenticación, de autorización y las combinaciones entre ellos.
- Valor del atributo: contiene los datos con los que se quiere interactuar.

En la figura 4 se expone de manera gráfica el formato de los atributos.

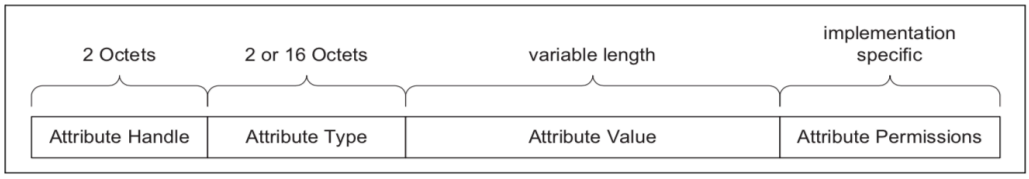

Fig. 4. Formato de los atributos (tomada de [5])

GATT

El perfil GATT está diseñado para emplearse en la comunicación de un cliente con el servidor a través de una aplicación u otro perfil. Como se ha dicho en los apartados anteriores, el GATT se construye por encima del protocolo ATT, por lo que se encarga de definir cómo utilizarle para el proceso de conexión de dos dispositivos BLE, es decir, cómo leer, escribir y obtener notificaciones de sus atributos. Para llevar a cabo esto, el perfil especifica los elementos de la estructura de datos contenida en el *Attribute Value* del protocolo ATT.

Un perfil está compuesto por uno o más servicios, que son colecciones de datos que describen un comportamiento o rol que permite la interoperabilidad de forma específica entre los dispositivos Bluetooth. Existen dos tipos de servicios: primarios y secundarios,

los primarios describen las funcionalidades principales del dispositivo, mientras que los secundarios sirven para dar apoyo a los primarios. Dentro de los servicios están las características, que contienen los datos que los servicios utilizan para cada función. Cada característica incluye unas propiedades que explican cómo se puede acceder a los datos y cómo pueden ser representados.

En la figura 5 se expone de forma gráfica esta jerarquía GATT:

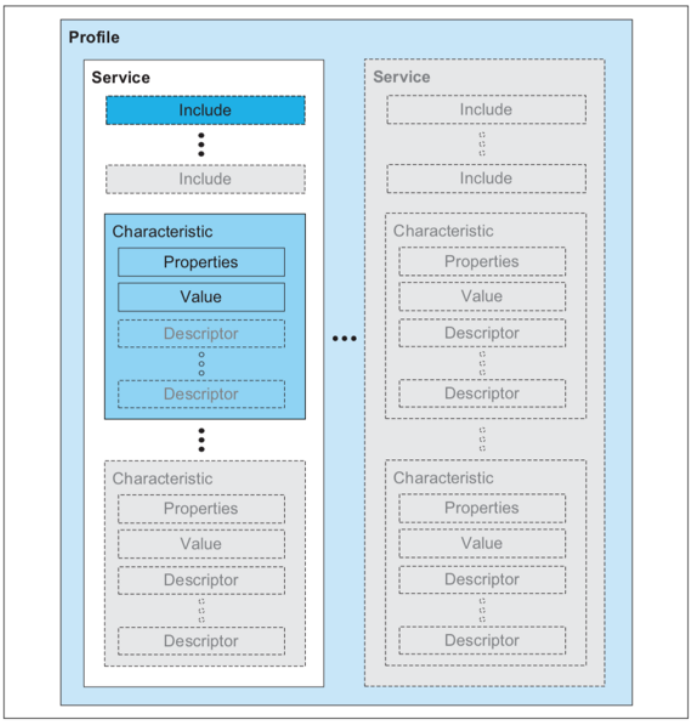

Fig. 5. Jerarquía de los perfiles GATT (tomada de [5])

Para entender mejor la jerarquía de los perfiles GATT, se expone a continuación uno de los muchos perfiles GATT que Bluetooth reconoce como oficiales [6], en concreto el relacionado con las notificaciones de alerta que se envían desde un teléfono inteligente a otro dispositivo, como puede ser un reloj.

Este perfil está compuesto por un servicio, que a su vez está compuesto por cinco características. Cada una de ellas se encarga de una función necesaria para llevar a cabo la comunicación y tiene unas propiedades predefinidas que permiten el flujo de mensajes entre los dispositivos. Por ejemplo, el valor de la característica llamada “Supported New Alert Category” es una serie de bits que, según estén a cero o a uno, determinan las propiedades que soporta la nueva alerta registrada.

En la figura 6 se observan las distintas características:

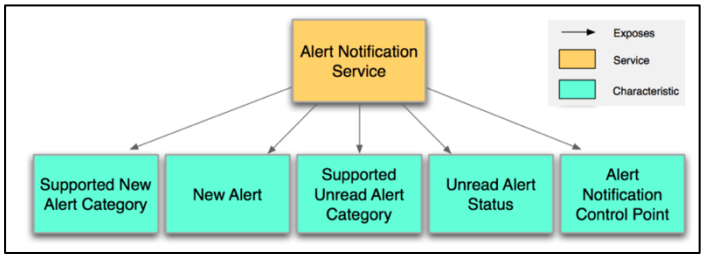

Fig. 6. Esquema del perfil de notificaciones de alerta (tomada de [18])

2.3. Protocolos de autenticación

Todo sistema de control de acceso necesita realizar una autenticación entre los dispositivos que garantice que uno de ellos, en este caso el teléfono móvil del empleado, está correctamente identificado y así se autorice su acceso a la infraestructura que protege el control. Concretamente, el proceso de autenticación se lleva a cabo mediante el envío de un desafío desde uno de los dispositivos, el "autenticador", al otro, que es el que se quiere autenticar, y dependiendo de la respuesta de este, la autenticación será correcta o no. Este proceso puede realizarse de muchas formas, incluyendo más o menos complejidad.

Como referencia se va a tomar el estándar definido en la RFC 3748 llamado *Extensible Authentication Protocol (EAP)* [7] que incluye una gran variedad de métodos de autenticación.

El funcionamiento de este protocolo es el siguiente:

- El "autenticador", que en este estándar coincide con el servidor, envía una petición al cliente. En esta petición, se incluye un identificador de la petición y el tipo de autenticación que se llevará a cabo.
- El cliente le envía el mensaje de respuesta, haciendo coincidir, tanto el identificador como el tipo de autenticación del servidor. Esto sirve para identificar conexiones y asegurar que los dos dispositivos utilizarán el mismo método en los siguientes mensajes.
- El intercambio de información continúa hasta que el "autenticador" considera que el cliente se ha autenticado correctamente. Esto depende de los mecanismos que se requieran en cada tipo de autenticación.

En los siguientes apartados, se analizarán dos de los protocolos más adecuados para el control de acceso. Para más información y otros ejemplos de protocolos de autenticación véase [8].

2.3.1. EAP Protected One-Time Password (EAP-POTP)

Este método, especificado en la RFC 4793 [9], consiste en la introducción de tokens OTP (One-Time Password, contraseñas de un solo uso) en el proceso de autenticación EAP. Estos tokens se generan a partir de una clave compartida o semilla, que es conocida por ambos extremos. Provee una autenticación del cliente frente al servidor en el modo simple, pero también puede proveer autenticación mutua si fuera necesario. El funcionamiento del modo simple se realiza de la siguiente forma:

- Los dispositivos se intercambian el identificador EAP mencionado anteriormente.
- El servidor envía una petición al cliente en la que incluye el tipo de autenticación, en este caso, EAP-POTP, el rango de versiones que soporta de este método, información útil del servidor, por ejemplo si está retomando una sesión iniciada anteriormente, y un campo llamado OTP TLV, que fuerza al cliente a responder con el OTP actual.
- El cliente, si la versión de su implementación se encuentra dentro del rango del servidor, comprueba si la sesión es nueva o se retoma una antigua. En este último caso, el cliente genera las nuevas claves de sesión y responde al servidor. En caso de ser una sesión nueva, calcula el token OTP a partir de la información que ha recibido del servidor más la clave compartida o semilla, lo encapsula en el campo OTP TLV y se lo manda junto a un campo que indique la versión del método más alta que soporta el cliente.
- Si la versión enviada por el cliente no coincide con la versión más alta del servidor, este envía una nueva petición con la versión soportada por el cliente. En caso contrario, comprueba el OTP recibido y decide si la autenticación es correcta o no. A partir de esto, le enviar un mensaje al cliente indicándole el resultado y se finaliza la conexión.

En la figura 7, se explica de forma gráfica el intercambio de mensajes de este protocolo.

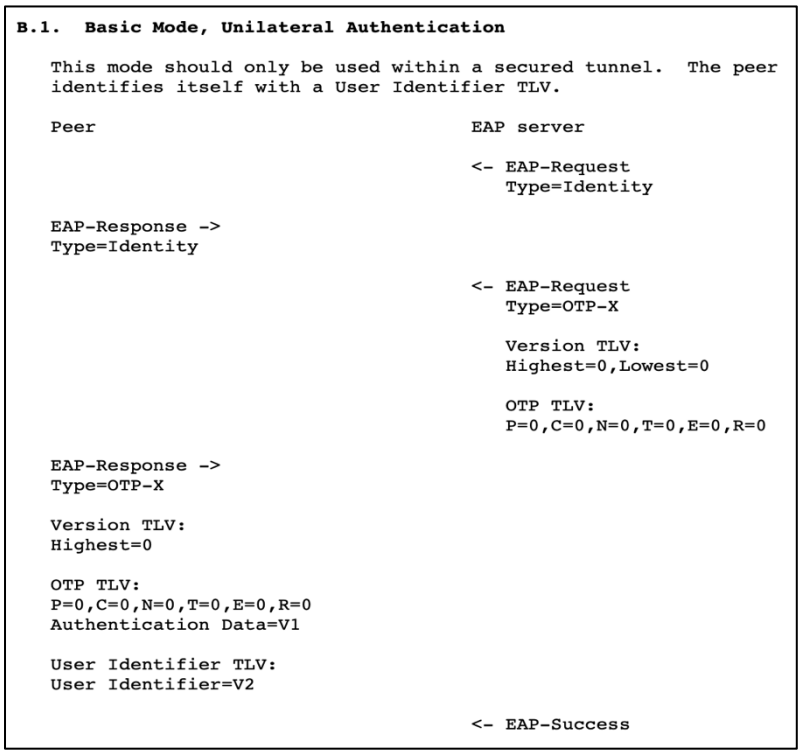

Fig. 7. Esquema del funcionamiento del modo simple EAP-POTP (tomada de [9])

2.3.2. EAP Pre-Shared Key (EAP-PSK)

Este método, especificado en la RFC 4764 [10], proporciona un canal de comunicación seguro gracias a la autenticación mutua de los extremos. Se basa en la derivación de claves estáticas y de sesión a partir de una clave que ambos dispositivos conocen antes de iniciar el protocolo, esto evita que se tengan que negociar los parámetros criptográficos a utilizar durante el intercambio de mensajes.

A partir de la clave compartida PSK (Pre-Shared Key) se calculan dos claves estáticas de larga duración, la AK (Authentication Key), que servirá para el proceso de autenticación mutua, y la KDK (Key-Derivation Key), que se utilizará para derivar tres claves de sesión compartidas entre cliente y servidor. Estas tres claves son: la TEK (Transient EAP Key), utilizada para establecer el canal protegido, la MSK (Master Session Key) y la EMSK (Extended Master Session Key).

En cuanto al canal seguro, este se establece para proteger las confirmaciones de la autenticación mutua, proporcionando integridad en los mensajes gracias al cálculo de una MAC (Message Authentication Code) con la clave TEK.

El funcionamiento de este protocolo se realiza de la siguiente forma:

- El servidor le envía un paquete al cliente en el que incluye el identificador de usuario de este y un número aleatorio generado en ese momento.
- El cliente procesa el mensaje identificando los parámetros recibidos y genera una MAC (Message Authentication Code) con la clave AK e incluyendo los parámetros recibidos, el identificador de usuario del cliente y un número aleatorio calculado por este. La MAC junto al número aleatorio y el identificador se envían al servidor.
- El servidor recibe la MAC del cliente y comprueba si es correcta o no, si lo es, el cliente se habrá autenticado y se procederá a establecer el canal seguro derivando las claves de sesión. Además, para proceder a la autenticación del servidor, este calcula otra MAC con la misma clave AK e incluyendo su identificador y el número aleatorio del cliente. Esta MAC se envía al cliente junto a la confirmación de la autenticación a través del canal seguro.
- El cliente comprueba que la MAC es correcta para autenticar al servidor y si lo es, manda la confirmación a través del canal seguro.

En la figura 8, se expone una gráfica que facilita la comprensión del intercambio de mensajes de este método.

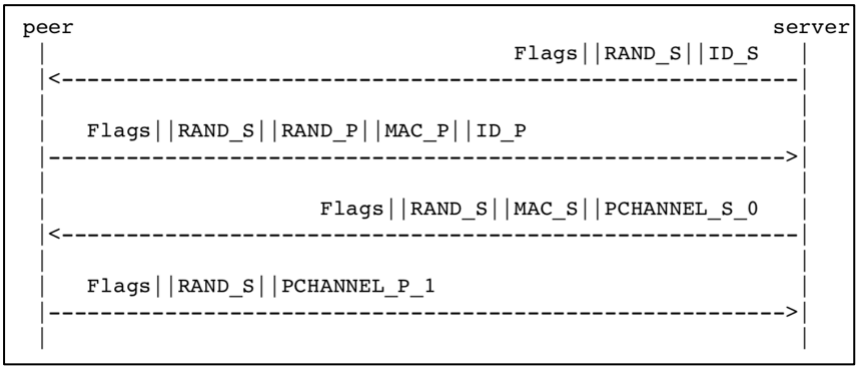

Fig. 8. Esquema de autenticación de EAP-PSK (tomada de [10])

2.4. Aplicación móvil

Actualmente, en el mercado de aplicaciones para dispositivos inteligentes destacan principalmente dos sistemas operativos, iOS y Android.

El sistema operativo iOS ha sido desarrollado por la compañía Apple Inc. y solo está disponible para los dispositivos fabricados por ella misma, es decir, no es compatible con teléfonos inteligentes o accesorios de otras marcas. Sin embargo, el sistema operativo Android ha sido desarrollado por la multinacional Google Inc. y su principal objetivo es la interoperabilidad entre marcas y dispositivos móviles.

Además, Android proporciona un "framework" completo que ayuda crear aplicaciones a partir de librerías ya implementadas por los desarrolladores de Android, y que permite iniciarse en el mundo de las aplicaciones móviles de manera sencilla [11] . Es por ello que para el desarrollo de la aplicación móvil de este proyecto se ha decidido utilizar este sistema operativo.

2.4.1. Aplicaciones Android

Android permite desarrollar aplicaciones desde la plataforma Android Studio creada por la misma empresa con el lenguaje de programación Java. Este software proporciona una interfaz parecida a otros programas de desarrollo software como Eclipse. Sin embargo, incluye características específicas como un editor de diseño visual y un emulador que ejecuta las aplicaciones sin necesidad de tener un dispositivo físico [12].

2.4.1.1. Componentes

Esta sección está basada en la referencia [13]. Los componentes son entidades individuales que actúan de manera independiente entre ellos y cumplen un rol específico dentro de las aplicaciones. Existen 4 tipos diferentes de componentes:

- Actividades: hacen referencia a una pantalla con interfaz de usuario, es decir, representan las pantallas en las que el usuario puede interactuar con la aplicación. Por ejemplo, las aplicaciones de mensajería instantánea tienen una actividad para mostrar los mensajes sin leer, otra para redactar nuevos mensajes, otra para leer las conversaciones individualmente, etc. Cada actividad es independiente de las demás, pero trabajan de manera conjunta para hacer funcionar la aplicación. De este modo, una aplicación puede iniciar

actividades de otras aplicaciones, como en el caso de la activación de la cámara de fotos en distintas aplicaciones.

- Servicios: hacen referencia a los componentes que se ejecutan en segundo plano y que realizan tareas prolongadas, por lo que no proporcionan interfaces de usuario. Son otros componentes, como las actividades, los que llaman a los servicios para iniciarlos.
- Proveedores de contenidos: administran los conjuntos de datos compartidos de las aplicaciones, la información que el sistema Android proporciona sobre el teléfono o el usuario, y los datos privados de las aplicaciones. Una vez que la aplicación adquiere los permisos necesarios para leer o escribir esa información, simplemente tiene que llamar a estos proveedores para que se la proporcionen.
- Receptores de mensajes: hacen referencia a los administradores que dan respuesta a los anuncios de mensajes en todo el sistema. Estos receptores son los encargados de crear las notificaciones que aparecen en la barra de estados que avisan al usuario cuando se produce un evento.

Cabe destacar que los cuatro componentes sirven como puntos de entrada o iniciadores de las aplicaciones, por lo que una aplicación puede tener varios puntos diferentes. Además, como se ha mencionado en el apartado de las "actividades", una aplicación puede activar componentes de otra distinta, lo que supone un punto de entrada distinto. Para llevar esto a cabo, se ha de crear un objeto "Intent" (derivado de la clase con el mismo nombre, que se decida a realizar enlaces entre componentes), a través del cual se avisa al sistema de que se quiere activar un componente de una aplicación externa, y este proceda a hacerlo. Esta utilidad será esencial para el correcto funcionamiento de la aplicación de este proyecto.

2.5. Conclusiones del capítulo

En este capítulo se han introducido las tecnologías y protocolos que se van a utilizar en el proyecto, así como las razones por las que se emplean.

A modo de resumen, cabe destacar la importancia de los elementos planteados que se van a utilizar en los capítulos siguientes: la placa de desarrollo elegida es la ESP32 DevKitC V4 debido a que sus características se adaptan a desarrollos profesionales, la tecnología para la conexión de los dispositivos es Bluetooth Low Energy gracias a su rapidez de respuesta y su interoperabilidad, el protocolo de autenticación es el EAP-PSK debido a

su sencillez y facilidad de implementación y, finalmente, el sistema operativo elegido para el desarrollo de la aplicación móvil es Android gracias a su compatibilidad con la mayoría de modelos de teléfono inteligente del mercado.

3. DISEÑO DE LA SOLUCIÓN

3.1. Descripción

En este apartado se va a exponer el diseño del sistema de control de acceso. Antes de describir la funcionalidad de cada componente en particular, es necesario tener una visión general del sistema de control de acceso al completo.

La secuencia de actividades que sigue el sistema es la siguiente, y queda representada en la figura 9:

1. El teléfono solicita acceso a la base de datos donde se encuentran los datos del empleado y los datos generales de la aplicación.
2. Se realiza una copia de los datos necesarios de la base de datos en el teléfono.
3. La placa de desarrollo está continuamente anunciándose por lo que el teléfono, una vez identificada la placa, le manda la petición de conexión.
4. La placa de desarrollo acepta la conexión y comienza el protocolo de autentificación EAP-PSK, enviando los datos correspondientes al primer mensaje del protocolo.
5. El teléfono móvil manda el segundo mensaje del protocolo, en el que se incluye la MAC que autentica al usuario.
6. La placa de desarrollo procesa esa MAC y dependiendo del resultado, abrirá la puerta o no. Esto se representa con el encendido del LED y del zumbador.

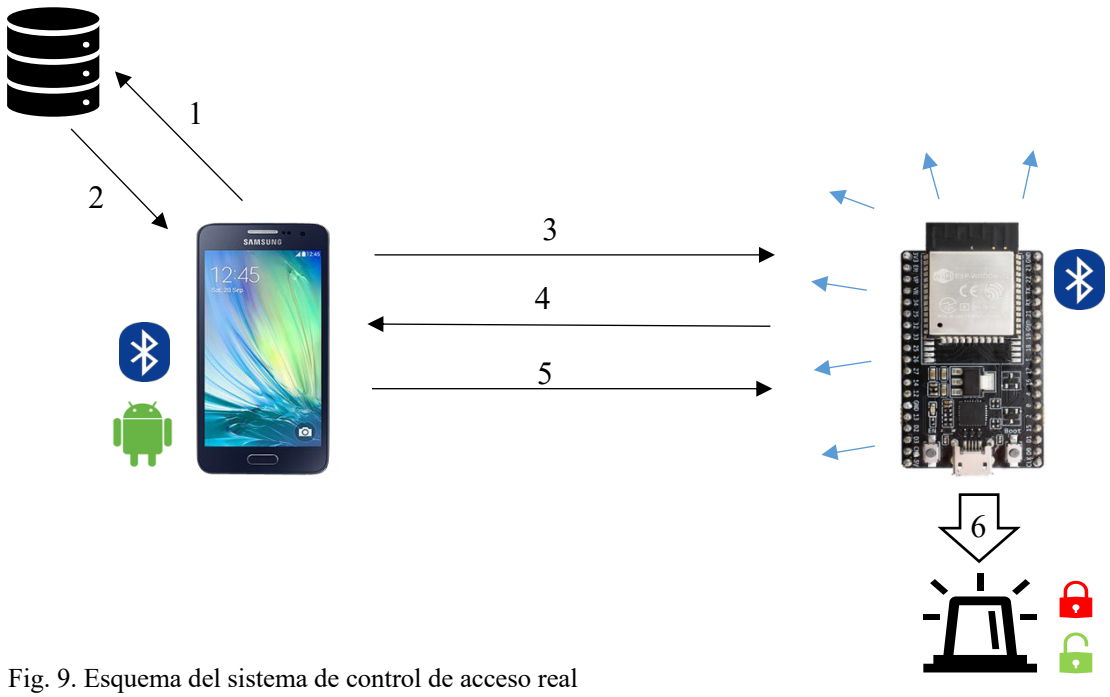

Fig. 9. Esquema del sistema de control de acceso real

3.1.1. Control de acceso

Como se ha mencionado anteriormente, para este proyecto se ha decidido emplear una placa de desarrollo ESP32 DevKitC V4 como dispositivo central de control de acceso. Además, para señalizar la autenticación correcta o incorrecta del usuario frente al control de acceso se hace uso de un LED tricolor RGB y un zumbador; estos accesorios simularían la apertura de la puerta o torno al que supuestamente se ha restringido el paso mediante el control de acceso.

Cuando la autenticación es correcta, el LED se ilumina en color verde y el zumbador emite un sonido durante 500 milisegundos, mientras que, si es incorrecta, el LED se ilumina en color rojo y el zumbador emite un sonido distinto durante los mismos segundos. En las figuras 10 y 11 se observa la iluminación del LED cuando la autenticación es correcta e incorrecta respectivamente.

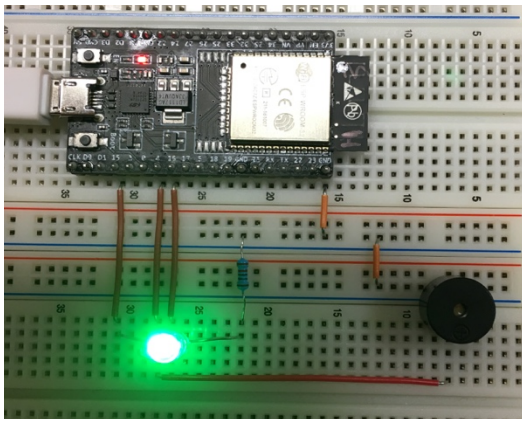

Fig. 10. Autenticación correcta del control de acceso

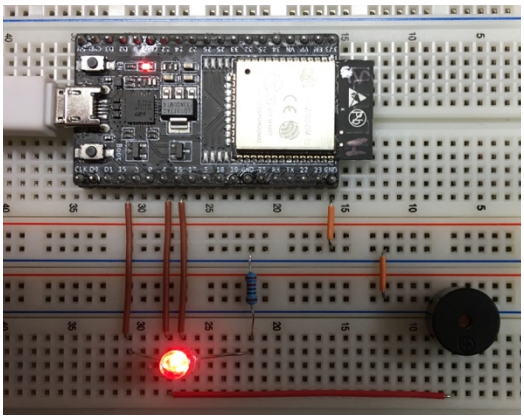

Fig. 11. Autenticación incorrecta del control de acceso

En cuanto a la conexión de los materiales, el LED está conectado en serie con dos pines de la placa de desarrollo, que son el 15 y el 16. El 15 hará que el LED luzca verde y el 16 hará que luzca rojo. A su vez, el cátodo está conectado a una resistencia de 220 Ohmios que evita la sobrecarga de voltaje, y esta, está conectada al GND (tierra) de la placa para cerrar el circuito y que se produzca el encendido de este. Respecto al zumbador, la parte positiva está conectada al pin 4 de la placa y la parte negativa al GND por el mismo motivo que el LED. En este caso no se le añade una resistencia puesto que el zumbador es capaz de trabajar a varios voltajes y no existe posibilidad de sobrecarga.

En la figura 12 se expone el diseño completo del control de acceso con sus respectivas conexiones.

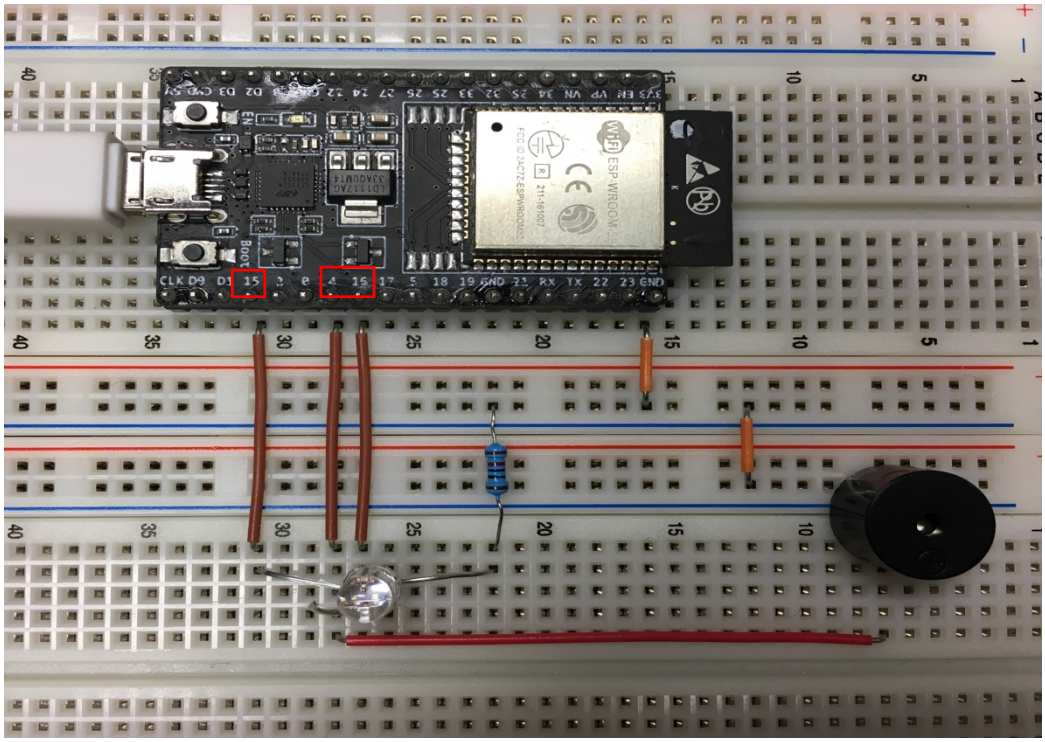

Fig. 12. Control de acceso

3.1.2. Aplicación Android

La aplicación Android que se ha creado para este proyecto recibe el nombre de AMB (Access Mobile Bluetooth) y es la encargada de actuar como "autenticador" frente al control de acceso, es decir, tiene la función de proveer la información necesaria sobre el usuario poseedor de la aplicación para que el control lo identifique como trabajador de la empresa y acepte su entrada.

A continuación, se describen las diferentes actividades que se desarrollan en la aplicación:

- Introducción de credenciales: se ha creado una tabla en la BBDD, en la que cada teléfono móvil corporativo está asociado a un solo empleado de la compañía gracias al IMEI (International Mobile Equipment Identity) del teléfono. Cuando el teléfono se conecta a ella, compara su IMEI con los de la base de datos y, una vez encontrado, identifica al usuario poseedor del móvil sin necesidad de registrarse. Esto es útil ya que evita que móviles ajenos a la compañía puedan utilizar la aplicación y, por tanto, personas ajenas puedan autenticarse.

En la figura 13 se observa la pestaña lateral de la aplicación en la que se incluyen los datos identificativos de cada empleado, que han sido descargados de la base de datos.

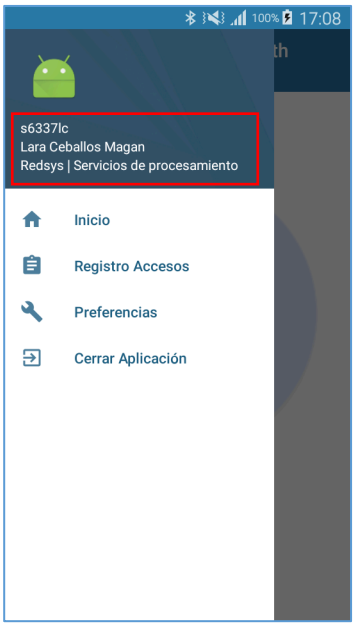

Fig. 13. Pestaña lateral con los datos del usuario

- Búsqueda de dispositivos y conexión: la aplicación está implementada como un servicio que trabaja en segundo plano, por lo que está constantemente escaneando en busca del dispositivo de control de acceso. La aplicación tiene un filtro de escaneo por el que busca solo los dispositivos con el nombre y la dirección MAC adecuados. Esta información está incluida en otra tabla de la base de datos, por lo que si se modificara algún dispositivo o se incluyera uno nuevo, la base de datos actualizaría la aplicación. Una vez que encuentra el dispositivo correcto, lanza una notificación por pantalla indicando si quiere conectarse y, por tanto, autentificarse. Pulsando la notificación, se acepta la conexión y se da comienzo al proceso de autenticación del usuario. En la figura 14 se muestra esta notificación.

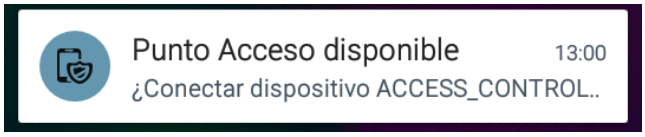

Fig. 14. Notificación de punto de acceso disponible

Aunque no es necesario que la aplicación esté abierta para que el servicio esté funcionando, en caso de que el usuario la tuviera encendida, en la pantalla se muestra un mensaje de "Buscando dispositivos…" mientras se estén escaneando los dispositivos Bluetooth de alrededor.

Fig. 15. Buscando dispositivos

Una vez encontrado el dispositivo, la imagen cambia de color y el mensaje pasa a ser "¿Conectar a dispositivo ACCESS_CONTROL_TFG?". Además, aparece un botón que tiene la misma función que la notificación de la barra de estado, acepta la conexión y el comienzo de la autenticación.

Fig. 16. Pregunta de aceptación de la conexión

Una vez pulsado el botón, el mensaje pasa a ser "Conectado a dispositivo: ACCESS_CONTROL_TFG" y la conexión queda establecida.

Fig. 17. Conexión establecida

- Registro de actividades: cada vez que el usuario se conecta a un control de acceso, la aplicación guarda ese proceso y lo muestra en la pestaña de "Registro Accesos". Así cada empleado puede acceder a su registro de entradas y salidas de las instalaciones y tener controlado el tiempo de trabajo. Para este proyecto, el prototipo solo consta de un control de acceso, que se ha decidido que sea de entrada, por lo que realmente no se puede contabilizar el tiempo, ya que se necesitaría uno adicional de salida. Como se puede observar en la figura 18, simplemente se muestra la fecha y hora en la que se produjo la entrada.

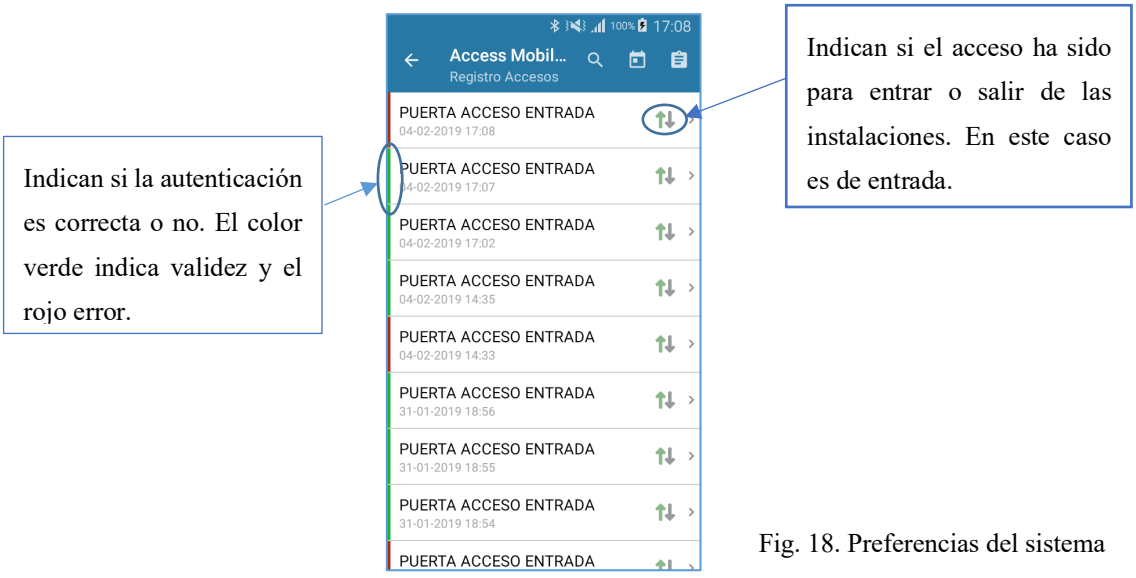

Fig. 18. Preferencias del sistema

- Preferencias del sistema: como toda aplicación móvil, incluye una pestaña de ajustes en la que se pueden modificar algunos aspectos de la aplicación. En el apartado de información general, se puede seleccionar si se desea que la aplicación empiece a funcionar cuando el teléfono se encienda; en la sección de notificaciones, se puede activar o desactivar el sonido y la vibración al recibir la notificación de conexión; y finalmente, en el último apartado, se incluye información útil sobre la aplicación. En la figura 19 se pueden observar las diferentes opciones mencionadas.

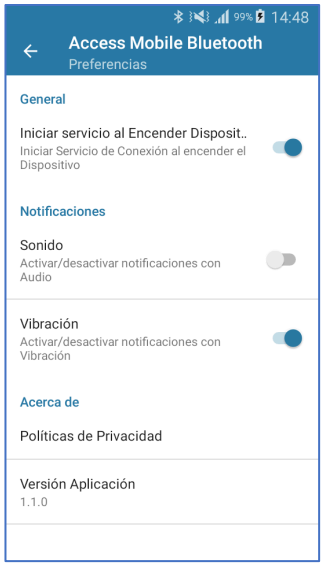

Fig. 19. Registro de actividades

- Cierre de la aplicación: existen dos opción para cerrar la aplicación. En la pestaña lateral existe la opción de "cierre de la aplicación" que se utiliza para salir de ella y finalizar la búsqueda de dispositivos en segundo plano. La figura 20 se resalta esta opción.

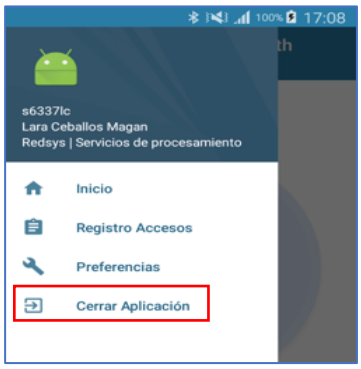

Fig. 20. Cierre de la aplicación

La segunda opción consiste en el cierre de la aplicación a través del botón de retroceso, lo que conlleva que el servicio de escaneo de dispositivos siga funcionando en segundo plano. En la figura 21 se muestra esta opción.

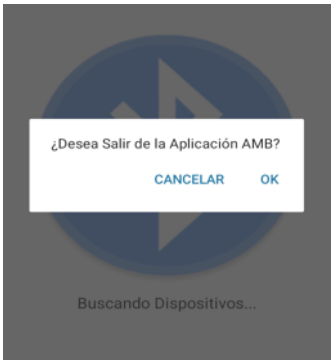

Fig. 21. Salir de la aplicación

3.1.2.1. Base de datos de la aplicación

Un punto importante de la aplicación es la base de datos, que está formada por 3 tablas que contienen:

- Los datos de los usuarios: nombre, identificador dentro de la empresa, empresa en la que trabaja, IMEI de su teléfono corporativo y clave compartida con el control de acceso.
- Los datos de los dispositivos de control: nombre, dirección MAC y alias, que se usará para mostrar en el registro de acceso.
- Los datos del registro de acceso: alias del dispositivo de control de acceso, fecha y hora del acceso, resultado de la conexión e indicador de entrada o salida.

Cuando la aplicación se inicia por primera vez, descarga su configuración y hace una copia de la estructura de la base de datos con las tablas vacías. Después cada componente de la aplicación que lo necesita accede a la base de datos original y copia los datos oportunos en las tablas vacías. Una vez realizado esto, cada vez que se requieren esos datos se accede a la base de datos del teléfono, lo que evita que se hagan accesos continuos a la base de datos situada en la nube.

La base de datos original es actualizada cada vez que hay un cambio en el registro de usuarios o en el registro de dispositivos de acceso, por ejemplo, cuando se incorporan empleados/dispositivos de control nuevos a la empresa, cuando algún miembro cambia de teléfono móvil por motivos justificados o cuando alguien deja la compañía. Cuando se

produce una modificación de estas, la base de datos notifica a la aplicación para que conecte con ella y se descargue los cambios producidos. Esto se realiza subiendo la versión de la aplicación en su configuración, lo que implica que el teléfono, al darse cuenta de que no tiene la última versión de la aplicación instalada se conecta con el servidor para descargársela. Así se garantiza que la aplicación tiene la base de datos actualizada en todo momento.

3.2. Análisis de seguridad

3.2.1. Seguridad en el sistema

En este apartado se exponen los requisitos de seguridad que el sistema de control completo busca proporcionar.

TABLA 2. REQUISITOS DE SEGURIDAD

Id. Requisito	Descripción
RF0001	La aplicación móvil debe autenticar al usuario frente al control de acceso mediante un protocolo seguro y robusto.
RF0002	El sistema debe estar protegido frente a ataques de repetición.
RF0003	El sistema debe estar protegido frente a ataques de diccionario.
RF0004	El sistema debe estar protegido frente a ataques Man-in-the-middle.
RF0005	La aplicación móvil debe protegerse frente al robo de datos.
RF0006	La aplicación móvil debe impedir que se pueda acceder a la información confidencial que maneja.

3.2.2. Seguridad en la conexión Bluetooth

En las últimas actualizaciones de Bluetooth se han implementado mecanismos para proporcionar seguridad en la conexión entre dispositivos. Sin embargo, en lo que respecta a BLE, esta seguridad es todavía menor que en los otros protocolos debido a su necesidad de emplear menos energía.

Existen 4 modos de funcionamiento dependiendo del nivel de seguridad que se emplee [5]:

- Funcionamiento simple (*Just Works*): funciona prácticamente igual que las versiones anteriores de Bluetooth, los usuarios simplemente tienen que aceptar la conexión. Por tanto, no proporciona ningún mecanismo contra ataques MITM (Man-in-the-middle).

- Comparación numérica (*Numeric Comparison*): requiere que los dos dispositivos que se conectan tengan pantalla o la posibilidad de conectarse a una, ya que se basa en mostrar un número aleatorio de 6 dígitos en ambos dispositivos a la vez y hacer que los usuarios acepten ese número si es igual en los dos extremos, lo que evita que se produzcan ataques de MITM.
- Fuera de banda (*Out of Band*): está diseñado para que se disponga de un dispositivo o mecanismo externo a Bluetooth que lleve a cabo el intercambio de la clave usada para la conexión mediante Bluetooth. Por tanto, se delega en un agente externo la seguridad frente a ataques.
- Entrada de contraseña (*Passkey Entry*): consiste en mostrar un número aleatorio de 6 dígitos en unos de los dispositivos e introducirlo en el otro, por lo que requiere que uno de los dispositivos tenga la capacidad de escribir texto por teclado. En cuanto a la seguridad, aporta la misma que el modo *Numeric Comparison*.

A pesar de tener 3 modos seguros, se ha demostrado que no son fiables ya que existen multitud de herramientas disponibles en la web que permiten seguir realizando ataques pasivos y de MITM durante la conexión como los expuestos en [14]. Debido a esto, se ha decidido utilizar un protocolo de autenticación alternativo implementado sobre el modo *Just Works*. El protocolo que se ha elegido es el EAP-PSK, explicado en el capítulo *Estado del arte*, ya que proporciona los siguientes servicios de seguridad [10]:

- Autenticación mutua: se consigue gracias al cálculo y verificación de MACs tanto en el cliente como en el servidor. Además, la clave usada para generar los MACs es exclusiva para esta tarea, lo que aporta independencia criptográfica al proceso.
- Integridad: se implementa a través del canal protegido, el cual encripta la información enviada y la protege calculando otro MAC con clave. Sin embargo, en la solución de este proyecto, no se incluye el canal protegido ya que no es necesario para un control de acceso, por lo que este servicio no queda cubierto.
- Protección frente a ataques de reinyección (Replay Attacks): durante la autenticación ambos dispositivos generan números aleatorios que introducen en sus mensajes y en la MAC, lo cual impide al atacante copiar el mensaje y reproducirlo posteriormente para engañar a uno de los participantes en la comunicación. Esto también ocurre en el canal protegido ya que los mensajes

que se mandan sobre este incluyen un campo que se va incrementando por cada mensaje enviado.

- Resistencia a ataques de diccionario: este protocolo no está basado en contraseñas que sean manejadas por humanos, por lo que no es susceptible a ataques de este tipo. De todos modos, para asegurar que esta protección sea posible, la PSK, que es la clave a partir de la que se derivan las demás, no debería generarse a partir de una contraseña, sino de un número aleatorio. Esto es justo lo que se ha implementado en el sistema de control.

3.2.3. Seguridad en el teléfono móvil Android

Actualmente, el mayor problema de seguridad que existe en un teléfono móvil es el robo de los datos confidenciales, ya que en él se guardan datos personales, contraseñas, datos bancarios o fotografías, por lo que ha de asegurarse que toda esta información se guarda en un lugar seguro. Para ello, Android ofrece distintos los siguientes tipos de almacenamiento según las necesidades de cada aplicación móvil [15]:

- Preferencias compartidas: permite el almacenamiento privado de pares clave-valor persistentes de tipos de datos primitivos: booleanos, strings, elementos flotantes y valores enteros.
- Almacenamiento interno: permite guardar en el almacenamiento interno del dispositivo archivos privados que solo son accesibles por la aplicación que los almacena, es decir, ni el usuario ni otras aplicaciones tienen acceso a ellos.
- Almacenamiento externo: permite guardar archivos en el almacenamiento externo del dispositivo, que puede ser una tarjeta microSD o una partición del almacenamiento interno. Estos archivos son accesibles por el usuario, el cual puede modificarlos, y también por otras aplicaciones, que pueden hacer operaciones de lectura y escritura.
- Bases de datos SQLite: permite el almacenamiento de datos estructurados en una base de datos privada, es decir, solo se puede acceder a ella desde la aplicación que la ha creado.
- Conexión de red: permite almacenar y obtener datos en la Web mediante servidores de red creados por el desarrollador de la aplicación.

Por otra parte, para incrementar la seguridad, Android sigue el *Principio de mínimo privilegio*, que consiste en dar accesos personalizados a cada aplicación y solo permitirles el acceso a las partes del sistema que son estrictamente necesarias para su funcionamiento.

Si alguna aplicación quiere acceder a otras partes como al almacenamiento externo del teléfono o los mensajes de texto, debe pedir su acceso explícito al usuario [13].

En el desarrollo de la aplicación de control de acceso se han empleado el almacenamiento interno y la base de datos SQLite privada para el almacenamiento de los datos del usuario, de los dispositivos de control, las claves de autenticación y los datos de registro que guarda la aplicación, por lo que no es posible acceder a esa información desde otras aplicaciones, ni desde el propio teléfono móvil. Además, para proporcionar un nivel mayor de seguridad, la base de datos está cifrada.

Para evitar la suplantación de identidad, como se ha indicado anteriormente, se ha asociado el IMEI del teléfono móvil con el usuario, por lo que solo los móviles entregados por la empresa son adecuados para instalarse la aplicación y proceder a la autenticación. Además, una vez que la aplicación está funcionando, el usuario debe aceptar la conexión explícitamente para poder autenticarse frente el control de acceso, lo que evita el ataque de MITM.

3.3. Conclusiones del capítulo

Este capítulo se ha divido en dos secciones. En la primera sección se ha explicado el diseño a alto nivel de los componentes que forman el sistema de control de acceso, es decir, del propio control de acceso formado por la placa de desarrollo, un LED y un zumbador; y de la aplicación móvil que hace la función de “autenticador”.

En la segunda sección, se ha desarrollado un análisis de seguridad, incluyendo los requisitos que deben cumplir, tanto la conexión Bluetooth como la aplicación Android, y cómo los han afrontado. Estos requisitos son importantes debido al propio objeto de este proyecto, que es la autenticación del usuario para acceder a un sitio restringido, esta autenticación debe producirse en un entorno seguro que evite robos de información y suplantación de identidad principalmente.

4. IMPLEMENTACIÓN DE LA SOLUCIÓN

4.1. Lista de requisitos

4.1.1. Requisitos funcionales

En este apartado, se exponen los requisitos funcionales para el sistema de control de acceso.

La tabla 2 representa los requisitos funciones que la aplicación móvil debe cumplir.

TABLA 3. REQUISITOS FUNCIONALES DE LA APLICACIÓN MÓVIL

Id. Requisito	Descripción
RF0001	Al iniciar la aplicación Android, la pantalla principal debe mostrar un icono parpadeante que indique la búsqueda de dispositivos, es decir, de control de acceso.
RF0002	El servicio de escaneo y conexión de dispositivos debe estar activado siempre, aunque la aplicación no esté iniciada.
RF0003	Cuando se encuentre el dispositivo adecuado, en la pantalla principal debe aparecer un botón que acepte la conexión con el control de acceso y comience el proceso de autenticación.
RF0004	Si la aplicación no está abierta, cuando la aplicación móvil encuentre el control de acceso, debe lanzar una notificación a la barra de estados del dispositivo.
RF0005	Cuando el usuario pulse en la notificación, se debe aceptar la conexión con el control de acceso y el comienzo del proceso de autenticación.
RF0006	Cuando la conexión se ha aceptado, desde la aplicación o desde la notificación, en la pantalla principal de la aplicación aparecerá un icono y un mensaje que indiquen que los dispositivos están conectados.
RF0008	En la pantalla principal debe existir un botón que permita acceder al menú lateral.
RF0009	El menú lateral debe mostrar los datos del usuario (identificador, nombre completo y empresa a la que pertenece), la opción de volver a la pantalla principal, de entrar a ver el registro de acceso, de acceder a las preferencias de la aplicación y, por último, la opción de cerrar la aplicación.
RF0010	Se debe guardar en la base de datos el registro de conexiones a dispositivos, indicando la hora y día que se ha producido la conexión, el nombre del control de acceso al que se ha conectado y si la conexión se produjo correctamente o no.
RF0011	En la pantalla de preferencias debe darse la posibilidad de cambiar la activación del servicio, si quiere arrancarse al encender el móvil o al iniciar la aplicación.
RF0012	En la pantalla de preferencias debe darse la posibilidad de cambiar la forma de notificar que la aplicación ha encontrado el control de acceso para conectarse.
RF0013	La aplicación debe proporcionar la posibilidad de parar el servicio que se ejecuta en segundo plano escaneando los dispositivos Bluetooth cuando el usuario quiera.

La tabla 3 representa los requisitos funcionales que la placa de desarrollo debe cumplir.

TABLA 4. REQUISITOS FUNCIONALES DE LA PLACA DE DESARROLLO

Id. Requisito	**Descripción**
RF0001	La placa de desarrollo debe tener un perfil GATT, que contenga al menos un servicio con dos características.
RF0002	La primera característica tiene que tener la propiedad de “solo lectura”.
RF0003	La primera característica contendrá los datos del primer mensaje del protocolo de autenticación para que la aplicación móvil los lea.
RF0004	La segunda característica tiene que tener la propiedad de “solo escritura”.
RF0005	La segunda característica recibirá los datos del segundo mensaje del protocolo que envía la aplicación móvil.
RF0006	Al iniciar la actividad, se debe habilitar el controlador Bluetooth para que se active solo el Bluetooth Low Energy.
RF0008	Una vez activado Bluetooth Low Energy, la placa de desarrollo debe crear e inicializar el servicio con sus respectivas características.
RF0009	Después de iniciar el servicio, debe empezar a enviar anuncios de conexión automáticamente.
RF0010	Cada vez que se realice una conexión, los valores de las características deben estar inicializados a cero para evitar interferencias entre distintos procesos de autenticación.
RF0011	La placa de desarrollo debe mandar una señal de activación al LED y al zumbador cuando finalice el proceso de autenticación.
RF0012	Dependiendo de si el proceso de autenticación se ha finalizado con éxito o no, el LED debe lucir de un color diferente, verde si se ha realizado correctamente y rojo si ha sido incorrecto.
RF0013	Dependiendo de si el proceso de autenticación se ha finalizado con éxito o no, el zumbador debe emitir un sonido diferente, si se ha realizado correctamente, el sonido será más agudo (4000Hz), y si ha sido incorrecto, será más grave (2000Hz).

4.1.2. Requisitos no funcionales

En este apartado, se exponen los requisitos no funcionales que se deben cumplir para el sistema de control de acceso.

TABLA 5. REQUISITOS NO FUNCIONALES

Id. Requisito	Descripción
RFN0001	La aplicación Android solo funcionará si está instalada en un teléfono móvil que pertenezca a una lista restringida (los facilitados por la empresa).
RFN0002	La versión mínima del API que debe soportar el sistema operativo Android instalado en el teléfono móvil es la 21.
RFN0003	La placa de desarrollo debe estar enchufada a la corriente para recibir potencia ya que no tiene una batería independiente.
RFN0004	Se debe iniciar manualmente el programa en la placa de desarrollo cada vez que se enchufe a la corriente.
RNF0005	La aplicación Android debe tener permisos para acceder a la ubicación del dispositivo para poder utilizar Bluetooth Low Energy.
RFN0006	La aplicación Android debe tener permisos para acceder al almacenamiento interno del dispositivo para poder almacenar la información de la base de datos y el registro de actividades.
RFN0007	La aplicación Android debe tener permisos para acceder a la información del teléfono móvil y los contactos para poder obtener el IMEI.
RFN0008	Ambos dispositivos tienen que tener habilitado Bluetooth Low Energy.
RFN0009	La placa de desarrollo debe tener instalado el SDK con las librerías de Bluetooth Low Energy instaladas.
RFN0010	La placa de desarrollo debe tener habilitados al menos 3 pines de salida, para los LED y el zumbador.
RFN0011	Al comenzar la conexión, ambos dispositivos deben negociar el tamaño de la MTU (Maximum Transmission Unit) para ampliarlo a 200 bytes para evitar problemas en el envío de paquetes.
RFN0012	Para evitar que el rango de señal Bluetooth de la placa de desarrollo sea muy amplio, se ha de modificar la potencia para que emita la menor posible.

4.2. Descripción

En este apartado se da una visión completa de la implementación del sistema de control de acceso.

4.2.1. Implementación de la aplicación Android

Android, como norma general, utiliza el lenguaje de programa Java. Existe la posibilidad de incluir código en los lenguajes nativos C y C++ y que puedan interactuar con el código Java. Sin embargo, para la aplicación de este proyecto se ha optado por trabajar solo con el lenguaje Java.

La aplicación está dividida en las siguientes clases:

- SplashActivity.java: es la actividad que se encarga de iniciar la aplicación. Para ello, revisa que esta tenga los permisos necesarios para su correcto funcionamiento y que el Bluetooth esté activado. Además, comprueba si el servicio de escaneo y conexión está arrancado, para iniciarlo en caso de que no lo esté. Por último, accede a la base de datos para obtener la información del usuario correspondiente con el IMEI del teléfono corporativo y llama a la actividad "MainActivity" a través de un "Intent".
- MainActivity.java: es la actividad correspondiente a la interfaz principal en la que se realiza la función de búsqueda del dispositivo de control de acceso cuando la aplicación está abierta y llama al servicio de conexión cuando lo encuentra. También es la encargada de desplegar el menú lateral y llamar a las diferentes opciones de este a través de "Intent".
- ConfirmAccessActivity.java: es la actividad encargada de interactuar con el control de acceso y realizar el proceso de autenticación, es decir, descarga la clave compartida del usuario, prepara los mensajes a enviar durante la autenticación y realiza las funciones de lectura y escritura de las características. Una vez finalizado este proceso, registra la información en la base de datos.
- RegisterAccessActivity.java: es la actividad correspondiente a la interfaz de registro de accesos. Se encarga de recoger la información relativa a las conexiones a los dispositivos de control de acceso, que ha sido almacenada en la base de datos por los componentes que realizan esa acción, y mostrarla como se ha indicado en el apartado de diseño.
- DataBase.java: aunque la base de datos, que contiene la información de la aplicación y el usuario, se ha creado con el programa DB Browser for SQLite,

en esta clase se implementan las operaciones de conexión, obtención de datos, actualización y copiado de esta.

- BluetoothLeService.java: es un servicio, basado en las especificaciones de Android, en el que se encuentran las funciones que son llamadas por los otros componentes para realizar sus tareas. Entre estas funciones se encuentran la de conexión de dos dispositivos BLE, las de lectura y escritura de características y la de modificar el MTU.
- ConnectionService.java: es el servicio que se encarga de las funciones de búsqueda y conexión al dispositivo de control de acceso cuando la aplicación no está abierta, por lo que también realiza las llamadas al sistema para que lance la notificación de conexión de la barra de estados. Cuando el usuario pulsa la notificación y los dispositivos se conectan, este servicio llama a la actividad "ConfirmAccessActivity" para iniciar el proceso de autenticación.

Por otra parte, en relación con la base de datos descrita en el capítulo de *Diseño*, cabe destacar que en este prototipo cuando los componentes necesitan algún dato, se copian las tablas enteras originales en las vacías creadas en el teléfono. Esto no supone ningún problema en cuanto a las tablas del registro de acceso (originalmente vacía) y de los dispositivos de control, pero sí conlleva una debilidad en la tabla de usuarios. Sin embargo, esta solución se ha empleado por simplicidad en el prototipo, pero tal como se explica en el apartado de *Líneas futuras de trabajo*, la implementación definitiva solo leería los datos asociados al usuario del teléfono (identificado por el IMEI) y esta lectura tendría que hacerse solo después de haber autenticado al usuario por su cuenta de empresa.

Además, como se ha mencionado en la sección de *Análisis de seguridad*, la tabla de usuarios de la base de datos está cifrada para proteger su contenido y evitar el robo de información sensible como son los datos del usuario y la clave compartida. Aunque no se ha tratado en este prototipo, también sería posible añadir un cifrado individual al campo de la clave cifrada que es prácticamente lo más importante dentro del proceso de autenticación.

4.2.2. Implementación de la placa de desarrollo

Para el desarrollo del código en el ESP32, se ha utilizado el lenguaje de programación C, debido a que las librerías del SDK están escritas en este lenguaje y no son compatibles con otros.

El programa implementado consta de un solo archivo .c en el que se incluyen las librerías de Bluetooth Low Energy y las relativas al acceso a los pines de la placa de desarrollo. También están integradas las llamadas al protocolo de autenticación, pero esto se explicará con detalle en el siguiente apartado.

En relación con la jerarquía del perfil GATT, como se ha mencionado en la lista de requisitos funcionales, este perfil tiene un servicio que incluye dos características, cuyos UUID son 0xFF01 y 0xFF02. Este servicio es el que interactúa con la aplicación móvil y, para ello, hace uso de las dos características. Cuando el protocolo de autenticación forma el primer mensaje, lo almacena en la primera característica 0xFF01, que únicamente tiene la propiedad de lectura. Sin embargo, cuando la placa recibe los datos de autenticación de la aplicación móvil, los almacena en la característica 0xFF02, que tiene únicamente la propiedad de escritura. Esto se ha implementado así para evitar problemas de sobre escritura entre los datos de entrada y salida.

En cuanto a la definición de estas características, las librerías de BLE de la propia placa de desarrollo, proporcionan una estructura ya definida en la que se declaran las propiedades de las características, los datos que contienen y el servicio al que pertenecen. Cada servicio está declarado en una posición diferente dentro del perfil GATT, por lo que la asignación de características a servicios se hace a través de esa posición. Si dos características están incluidas en diferentes servicios, el valor de la posición será diferente y, al contrario, cuando estén en el mismo.

A continuación, se expone la declaración de las dos características, en la que se puede observar que ambas pertenecen al mismo servicio ya que la variable “service_pos” tiene el mismo valor. Además, los permisos son comunes para las dos ya que son necesarios para que la propia placa de desarrollo. Sin embargo, cada una tiene una propiedad distinta, la primera de lectura y la segunda de escritura, como se ha comentado anteriormente.

```
struct gatts_char_inst gatts_char[GATTS_CHAR_NUM] = {
       {
        .service_pos = 0,
             .char_uuid.len = ESP_UUID_LEN_16,
             .char_uuid.uuid.uuid16 = GATTS_CHAR_UUID_READ,
             .char_perm = ESP_GATT_PERM_READ | ESP_GATT_PERM_WRITE,
             .char_property = ESP_GATT_CHAR_PROP_BIT_READ,
             .char_val = &gatts_char1_val,
             .char_control = NULL,
             .char_handle = 0,
             .char_nvs = ""
       },
       {
        .service_pos = 0,
             .char_uuid.len = ESP_UUID_LEN_16,
             .char_uuid.uuid.uuid16 = GATTS_CHAR_UUID_WRITE,
             .char_perm = ESP_GATT_PERM_READ | ESP_GATT_PERM_WRITE,
             .char_property = ESP_GATT_CHAR_PROP_BIT_WRITE,
             .char_val = &gatts_char2_val,
             .char_control = NULL,
             .char_handle = 0,
             .char_nvs = ""
       },

};
```

Por otra parte, para manejar todos los eventos que BLE manda al programa, se hace uso de dos funciones relacionados con los protocolos GAP y GATT. La primera función, llamada “gap_event_handler()”, se encarga de manejar los eventos relativos a los anuncios que se lanzan desde la placa de desarrollo y a la búsqueda de dispositivos. Mientras que la segunda función, llamada “gatts_event_handler()” está relacionada con la configuración de los servicios y características, así como con la interacción del usuario con estos.

En cuanto a la minimización de la señal Bluetooth mencionada en los requisitos no funcionales, se hace uso de una función en las librerías de la placa ESP32 que permite regular la potencia de la señal y que se declara como:

```
esp_ble_tx_power_set(esp_ble_power_type_t power_type, esp_power_level_t power_level)
```

El primer argumento se corresponde con el tipo de actividad a la que se quiere modificar la potencia y el segundo hace referencia al nivel de potencia que se quiere alcanzar. Este método se llama justo después de habilitar el Bluetooth de la placa.

Después de la activación de esta función, comienza el flujo correspondiente a la inicialización del servicio y las características que se detalla en la figura 22.

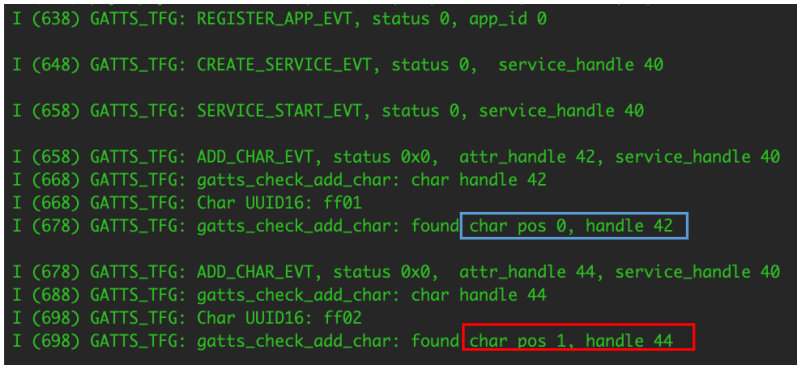

Fig. 22. Proceso de inicialización del control de acceso

Primero, el “REGSITER_APP_EVT” registra el perfil GATT y llama a las funciones de anuncio y escaneo correspondientes al protocolo GAP. A continuación, se crea el servicio definido y se arranca en los eventos “CREATE_SERVICE_EVT” y “SERVICE_START_EVT”. Una vez, finalizada la inicialización del servicio, se llama a “ADD_CHAR_EVT” que añade las dos características al servicio.

A continuación, la figura 23 muestra el proceso que sigue la placa de desarrollo cuando se conecta al dispositivo móvil.

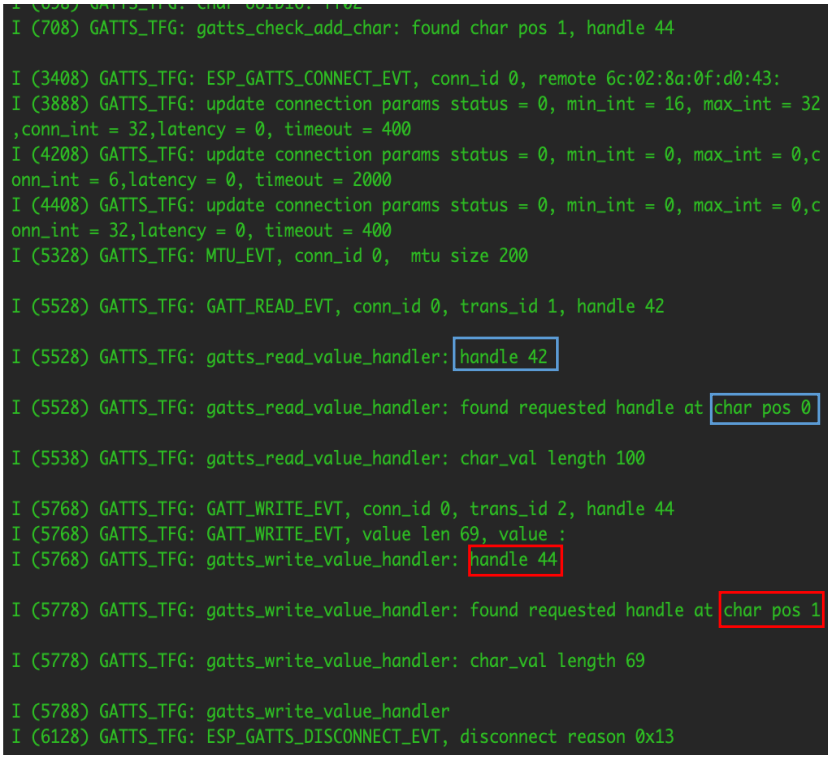

Fig. 23. Proceso de conexión del control de acceso

Cuando la aplicación móvil encuentra la placa de desarrollo, la envía automáticamente la petición de conexión y esta la acepta en el evento "ESP_GATT_CONNECT_EVT", donde también se actualiza la información de conexión. En la misma petición de conexión, la aplicación Android requiere que la MTU sea 200, por lo que la placa lanza el evento "MTU_EVT" para confirmarlo. Una vez establecida la conexión, se procede a la lectura y escritura de los mensajes de autenticación a través de los eventos "GATTS_READ_EVT" y "GATTS_WRITE_EVT". Se puede observar en las figuras 22 y 23 que el evento de lectura se hace a través de la primera característica ya que el "handler" y la posición de la característica coincide con esta. Lo mismo pasa en el evento de escritura que se hace a través de la segunda característica. Por último, cuando el proceso de autenticación finaliza, los dispositivos se desconectan automáticamente, y en el caso de la placa de desarrollo se llama al evento "ESP_GATTS_DISCONNECT_EVT".

4.2.3. Implementación del protocolo de autenticación

Como ya se ha mencionado en varios capítulos anteriores, el protocolo de autenticación en el que está basada la implementación de este proyecto es el EAP-PSK. Sin embargo, en vez de incluir una autenticación mutua, se ha simplificado a una autenticación simple del usuario frente al control de acceso para mejorar la experiencia del usuario. Eso se consigue debido a que se intercambian menos mensajes, solo dos, y así se aumenta la velocidad de apertura de la puerta. En el esquema de la figura 24 quedan reflejados los mensajes intercambiados especificando su contenido.

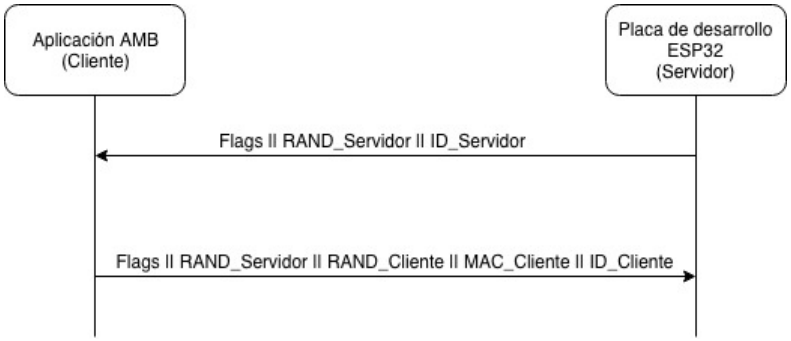

Fig. 24. Esquema del protocolo de autenticación

El primer mensaje es generado por la placa de desarrollo y está compuesto por:

- Flags: ocupan 6 bytes divididos en la posición del mensaje en el flujo, un identificador genérico EAP, la longitud del mensaje, el tipo de método EAP utilizado, en este caso EAP-PSK que es el 47 (2F en hexadecimal), y el identificador del mensaje EAP-PSK.
- RAND_Servidor: es un número aleatorio de longitud 16 bytes, calculado dentro del protocolo de autenticación.
- ID_Servidor: es un identificador único del control de acceso de 9 bytes.

El segundo mensaje es generado por la aplicación móvil y está compuesto por:

- Flags: formado por los mismos elementos que en el primer mensaje.
- RAND_Servidor: obtenido a partir del primer mensaje enviado por la placa y recibido en la aplicación Android.
- RAND_Cliente: es un número aleatorio de longitud 16 bytes calculado con la clase SecureRandom de Java.
- MAC_Cliente: es una MAC calculada a partir los datos que se recogen de la placa de desarrollo (RAND_Servidor e ID_Servidor) y los generados en la aplicación Android (RAND_Cliente e ID_Cliente). En el protocolo EAP-PSK se especifica que la clave con la que calcular el MAC debe derivarse de una clave compartida por los dos extremos antes de iniciar la conexión. Sin embargo, la criptografía de Android es limitada y no da la posibilidad de diversificar las claves con el mismo método que emplea EAP-PSK, por lo que la clave utilizada para la MAC es directamente la clave compartida previamente entre los dispositivos. Por esta razón también se ha tenido que cambiar el método de cálculo de las funciones de resumen, inicialmente OMAC, ya que Android solo las calcula con el método HMAC y debido a la limitación de 16 bytes que impone el protocolo EAP-PSK, el único disponible era HMAC-MD5.
- ID_Cliente: es un identificador único de 15 bytes que se corresponde con el IMEI del teléfono móvil. De este modo se asegura que sea único para cada usuario.

Una vez escrito este mensaje en la placa de desarrollo, esta tiene que validar que la MAC recibida es correcta. Para ello, primero busca la clave compartida con el usuario a través del ID_Cliente. Se ha creado una estructura en la que cada ID_Cliente está asociado con una clave, por lo que cada usuario tiene su propia clave única compartida con el control

de acceso. Una vez, que la placa encuentra la clave asociada, agrupa los datos incluidos en la MAC (los dos números aleatorios y los identificadores) y con esa clave calcula su propia función de resumen. Si las dos MAC resultan ser la misma, la validación será correcta y, por tanto, la autenticación del usuario también. Si en este proceso la placa encuentra algún dato incorrecto, ya sean las longitudes de las variables o los propios datos recibidos, automáticamente se daría por incorrecta la autenticación.

En cuanto a la generación de la clave compartida mencionada en el cálculo de las MAC, se ha empleado el comando siguiente (disponible para Linux y MacOS) para calcularla:

```
head -c 4096 /dev/urandom | openssl sha256 | cut -b1-16
```

Este comando emplea la utilidad para calcular números pseudoaleatorios (/dev/urandom) que tienen estos sistemas operativos, y el resultado lo envía a la herramienta openSSL para que calcule una función de resumen de tipo "sha256" y devuelva 16 bytes, que son los requeridos por el protocolo EAP-PSK. Este comando se ha obtenido de la referencia [16].

Una vez que se ha generado la clave, se introduce, asociándola al IMEI de cada usuario, en el código de la placa de desarrollo y en la base de datos de la aplicación manualmente.

Cabe destacar que la implementación del protocolo EAP-PSK correspondiente a la parte de la placa de desarrollo, es decir, la construcción del primer mensaje y la validación del segundo está basada en las implementaciones realizadas por Jouni Malinen[1], que están distribuidas bajo la licencia "BSD license".

4.2.4. Efectos de las simplificaciones realizadas

En este apartado se exponen los efectos de las simplificaciones introducidas, sobre todo, en el protocolo de autenticación.

Este protocolo, como se sabe, puede proporcionar una autenticación mutua. Sin embargo, en este proyecto se ha simplificado a una única autenticación del usuario ante el control de acceso, por lo que se elimina la autenticación del servidor. Esto podría implicar una vulnerabilidad frente ataques MITM, pero para ello, primero, el servidor intruso debería

[1] La librería desde donde se puede acceder las implementaciones mencionadas es: https://android.googlesource.com/platform/external/wpa_supplicant_8/+/ics-plus-aosp/src

duplicar el nombre y la dirección MAC de uno de los controles guardados en la base de datos para que la aplicación lo reconociera. Después, si consiguiera eso, la información que el móvil envía realmente no se puede usar excepto inmediatamente, ya que el número aleatorio cambia en cada conexión. Además, el usuario se daría cuenta de que el control no le habría autorizado, por lo que un ataque de este tipo no sería posible en la práctica.

Por otra parte, como se ha explicado anteriormente se han realizado modificaciones en el cálculo de las MAC. Estas dos modificaciones suponen bajar el nivel de seguridad del protocolo debido a que, por una parte, al ser una clave compartida tiene más facilidad para ser interceptada en el envío a los dispositivos, y por otra, es conocido que el método MD5 tiene vulnerabilidades y está roto desde 2004. Sin embargo, en este prototipo las claves se introducen manualmente, por lo que no hay posibilidad de interceptarlas, y en cuanto a la función de resumen, bastaría con aumentar su longitud, ya que Android soporta HMAC-Sha224 o HMAC-Sha256.

4.3. Conclusiones del capítulo

En este capítulo se ha expuesto la implementación del sistema de control de acceso completo. Primero, se han elaborado tres tablas con los requisitos fundamentales que debe cumplir cada dispositivo del sistema expuesto para su correcto funcionamiento. A continuación, se ha explicado cómo se ha desarrollado la aplicación móvil, el programa que está integrado en la placa de desarrollo y el protocolo de autenticación que llevan a cabo los dispositivos. Por último, se han explicado los efectos que tienen las simplificaciones realizadas sobre el sistema de control.

5. PRUEBAS REALIZADAS

5.1. Prueba de autenticación correcta

El fin de esta prueba consiste en garantizar el buen funcionamiento del sistema de control de acceso en condiciones normales. Para ello se han realizado un total de 25 pruebas y en todas ellas los dispositivos se han autenticado con éxito. A efectos demostrativos, en este apartado se describe una de ellas.

Después de compartir la clave entre ambos extremos, se inicia la placa de desarrollo para que pueda empezar a enviar anuncios de conexión. Automáticamente después aparece una notificación en el teléfono móvil pidiendo la conexión con el dispositivo, que se muestra en la figura 25.

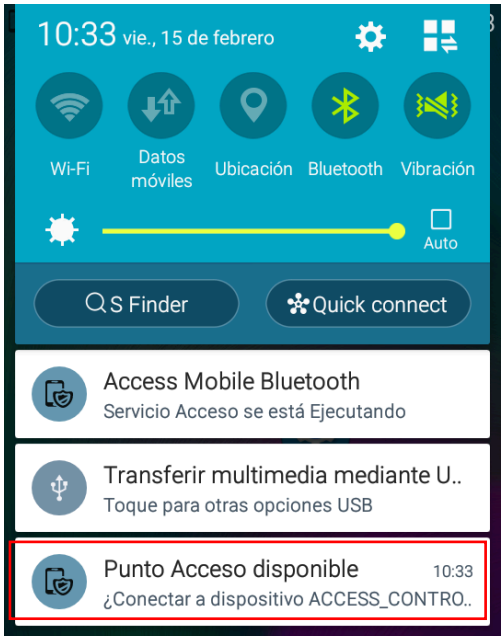

Fig. 25. Notificación de conexión durante la prueba

Pulsando en la notificación señalada, se da comienzo a la conexión y el intercambio de mensajes. Para la lectura del primer mensaje se hace uso de la función "readCharacteristic" que proporciona la clase *BluetoothGatt* de Android. Este mensaje tiene la forma siguiente, que coincide con la expuesta en el capítulo anterior:

- Flags: 0x01(número de secuencia del flujo de mensajes) 0x01 (identificador protocolo EAP) 0x00 0x1f (longitud del mensaje) 0x2f (identificador método EAP-PSK) 0x00 (número de secuencia necesario para el protocolo EAP).

- Número aleatorio generado por el servidor: 0xf7 0xd3 0x0e 0x56 0x62 0xbc 0x46 0xbc 0xfa 0xd1 0x3d 0x33 0x10 0x36 0x0d 0x69
- Identificador único del servidor: 0x70 0x72 0x6f 0x74 0x6f 0x74 0x79 0x70 0x65

Una vez que la aplicación lee este mensaje, procede a generar el segundo mensaje que escribirá, con la función "writeCharacteristic" de la misma clase *BluetoothGatt*, en la placa de desarrollo con la siguiente forma:

- Flags: 0x02 0x01 0x00 0x3f 0x2f 0x01 (misma división que en el primer mensaje).
- Número aleatorio del servidor: 0xf7 0xd3 0x0e 0x56 0x62 0xbc 0x46 0xbc 0xfa 0xd1 0x3d 0x33 0x10 0x36 0x0d 0x69 (coincide con el obtenido del primer mensaje).
- Número aleatorio generado en el cliente: 0x49 0xba 0xf5 0xbe 0x14 0x96 0x88 0x4e 0x21 0xa2 0x62 0x07 0xf3 0xee 0x4c 0xf6
- MAC generada por el cliente: 0xee 0x01 0x35 0xba 0x85 0x54 0x0c 0x4a 0x06 0x9d 0x73 0x11 0x21 0x94 0xd2 0x13
- Identificador único del cliente: 0x33 0x35 0x33 0x39 0x37 0x34 0x30 0x37 0x39 0x36 0x31 0x35 0x38 0x35 0x31 (coincide con el IMEI del teléfono, pero en formato ASCII).

A continuación, la placa de desarrollo procede a analizar el mensaje y generar su propia MAC para verificar la enviada por la aplicación móvil. En este caso, ambas funciones de resumen son iguales, por lo que la autenticación es correcta. Esto se demuestra en la figura 26 de la página siguiente, la cual muestra el proceso que sigue la placa después de la aceptación de conexión, es decir, la lectura y escritura de los dos mensajes en las características, y la validación de la MAC. Cabe aclarar que la función de resumen llamada "MAC" es la calculada por la placa de desarrollo y la llamada "MAC_P" es la obtenida del segundo mensaje.

Después de esta autenticación, se envía el resultado al LED, que se enciende en color verde y el zumbador que emite el sonido a 4000Hz. La figura que se corresponde con esto sería la misma que la figura 10.

```
I (5628) GATTS_TFG: GATT_READ_EVT, conn_id 0, trans_id 1, handle 42

I (5628) GATTS_TFG: gatts_read_value_handler: handle 42

I (5628) GATTS_TFG: gatts_read_value_handler: found requested handle at char pos
 0

PRIMER MENSAJE:
0x01 0x01 0x00 0x1f 0x2f 0x00 0xf7 0xd3 0x0e 0x56 0x62 0xbc 0x46 0xbc 0xfa 0xd1
0x3d 0x33 0x10 0x36 0x0d 0x69 0x70 0x72 0x6f 0x74 0x6f 0x74 0x79 0x70 0x65
I (5648) GATTS_TFG: gatts_read_value_handler: char_val length 100

I (5828) GATTS_TFG: GATT_WRITE_EVT, conn_id 0, trans_id 2, handle 44
I (5828) GATTS_TFG: GATT_WRITE_EVT, value len 69, value :
I (5828) GATTS_TFG: gatts_write_value_handler: handle 44

I (5838) GATTS_TFG: gatts_write_value_handler: found requested handle at char po
s 1

I (5838) GATTS_TFG: gatts_write_value_handler: char_val length 69

SEGUNDO MENSAJE:0x02 0x01 0x00 0x3f 0x2f 0x01 0xf7 0xd3 0x0e 0x56 0x62 0xbc 0x46
 0xbc 0xfa 0xd1 0x3d 0x33 0x10 0x36 0x0d 0x69 0x49 0xba 0xf5 0xbe 0x14 0x96 0x88
 0x4e 0x21 0xa2 0x62 0x07 0xf3 0xee 0x4c 0xf6 0xee 0x01 0x35 0xba 0x85 0x54 0x0c
 0x4a 0x06 0x9d 0x73 0x11 0x21 0x94 0xd2 0x13 0x33 0x35 0x33 0x39 0x37 0x34 0x30
 0x37 0x39 0x36 0x31 0x35 0x38 0x35 0x31
MAC_P
0xee 0x01 0x35 0xba 0x85 0x54 0x0c 0x4a 0x06 0x9d 0x73 0x11 0x21 0x94 0xd2 0x13
MAC
0xee 0x01 0x35 0xba 0x85 0x54 0x0c 0x4a 0x06 0x9d 0x73 0x11 0x21 0x94 0xd2 0x13
I (6268) GATTS_TFG: ESP_GATTS_DISCONNECT_EVT, disconnect reason 0x13
```

Fig. 26. Secuencia de mensajes correctos en la placa de desarrollo

Para finalizar la prueba, se revisa que el registro de acceso de la aplicación Android ha registrado esta conexión correctamente. La figura 27 expone esto.

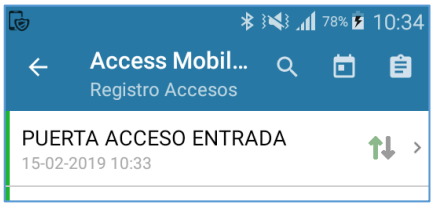

Fig. 27. Registro de acceso correcto

5.1.1. Prueba de velocidad

Una vez superada la prueba de autenticación correcta, se ha procedido a calcular la duración del proceso de conexión y autenticación. En las 25 pruebas anteriores se ha medido de forma repetida con un cronómetro el tiempo que se tarda en llevar a cabo estas acciones, desde la pulsación de la notificación en la barra de estados hasta la iluminación del LED y la emisión de sonido por parte del zumbador.

El resultado medio ha sido de 2,20 segundos con un intervalo de confianza de 0,15 segundos. En el anexo A se adjunta la tabla con los tiempos de cada prueba.

Por tanto, se puede concluir que la duración del proceso es baja, e igual o menor que con el mecanismo que actualmente se utiliza en la empresa, por lo que la experiencia del usuario resulta mejor.

5.2. Prueba de autenticación incorrecta

El objetivo de esta prueba es demostrar que el programa implementado en la placa de desarrollo es resistente al envío de datos incorrectos por parte de la aplicación móvil, ya sea porque no cumplan las longitudes correspondientes o porque la MAC está mal calculada.

En concreto, en esta prueba se va a simular que la aplicación no lee correctamente el número aleatorio del servidor y lo pone todo a ceros. Esto implica que la MAC se calcule de manera incorrecta y, por tanto, no coincida con la que calcula posteriormente la placa de desarrollo.

En la figura 28 de la página siguiente, se observa que los mensajes siguen el mismo esquema que en el apartado anterior, con la diferencia de que en el segundo mensaje los 16 bytes, que debería ocupar el número aleatorio del servidor, están todos ceros. Además, las dos funciones de resumen no coinciden debido a este problema.

Cuando la placa detecta este problema, automáticamente lo comunica al LED y al zumbador y la demostración sería la misma que la figura 11.

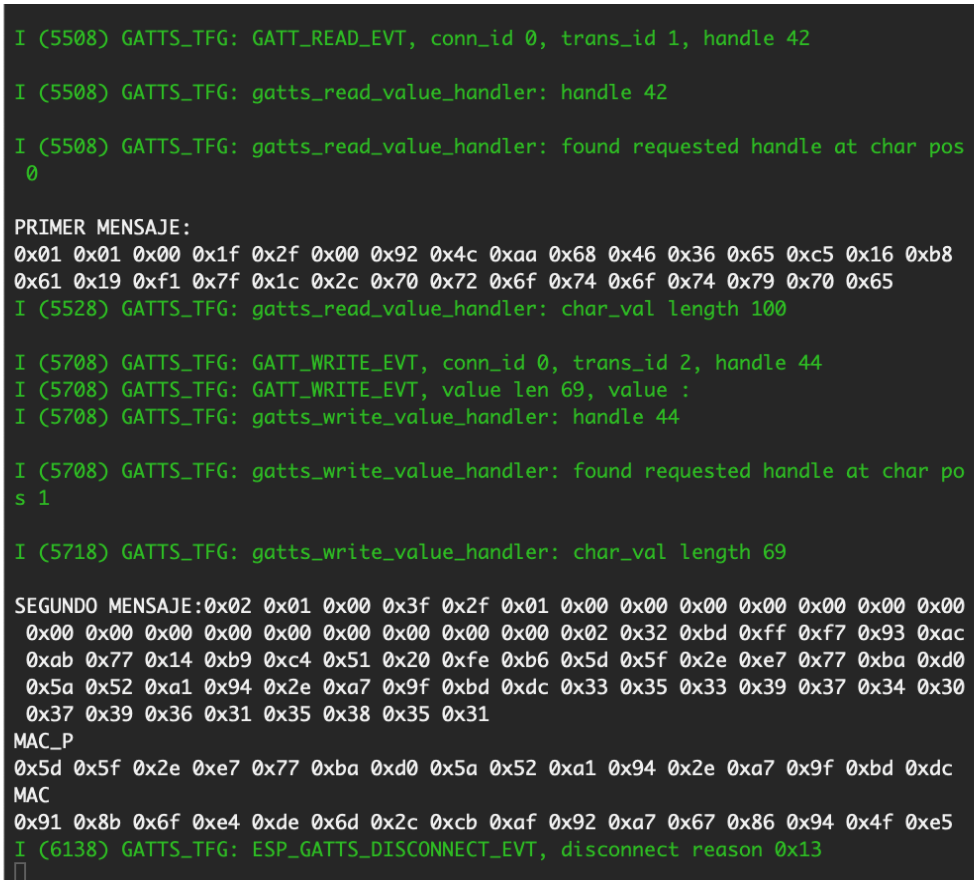

Fig. 28. Secuencia de mensajes incorrectos en la placa de desarrollo

5.3. Prueba de alcance de Bluetooth

El fin de esta prueba es comprobar que el área que abarca la señal de Bluetooth en la placa de desarrollo es la menor posible. Esta prueba es importante debido a que los controles de acceso de este proyecto están destinados a abrir puertas o activar tornos, y por tanto, permitir el acceso al edificio de la compañía. Si la señal Bluetooth saliera del edificio, es decir, a la calle, sería posible activar el acceso desde allí, lo que resultaría en un problema de seguridad.

Para evitar esto, se hace uso de la función explicada en el capítulo de *Implementación.* En este proyecto, el primer argumento se corresponde con los anuncios que lanza la placa y el segundo hace referencia al menor nivel de potencia que se puede declarar, que son -12dBm. Por tanto, la función queda implementada de la siguiente forma:

```
esp_ble_tx_power_set(ESP_BLE_PWR_TYPE_ADV, ESP_PWR_LVL_N12)
```

Una vez ejecutada la función, se procede a probar si la señal disminuye lo suficiente. Para ello, se ha dejado el control de acceso en un punto fijo y se ha ido distanciando el teléfono móvil progresivamente. A una distancia de aproximadamente 10 metros con obstáculos de por medio, el teléfono se sigue pudiendo conectar al control de acceso, aunque no lo detecta con la misma facilidad.

Por tanto, se concluye que, aunque la intensidad de la señal disminuye cuando se implementa esta función, el usuario sigue pudiendo autenticarse frente al control de acceso a una distancia excesivamente lejana, lo que supone una debilidad del sistema y requiere un mayor estudio en el futuro.

5.4. Prueba en teléfono no corporativo

El objetivo de esta prueba es comprobar que teléfonos ajenos a la empresa Redsys no pueden iniciar la aplicación y, por tanto, usuarios que no pertenezcan a ella no pueden vulnerar la seguridad del control de acceso.

Como se ha comentado anteriormente, la aplicación obtiene los datos del usuario de una base de datos SQLite. En esta base de datos el usuario está asociado con el IMEI del teléfono móvil, por lo que no es necesario emplear un "log-in" para obtener los datos del trabajador e iniciar la aplicación.

Actualmente, la tabla "USER_DATA" de la base de datos contiene dos usuarios de prueba. El primer usuario se corresponde con la persona que ha realizado este TFG y su IMEI pertenece al teléfono móvil utilizado para la implementación del proyecto. El segundo usuario es totalmente ficticio, pero sirve para demostrar que la aplicación soporta la búsqueda de usuarios cuando se incluyen varios en la base de datos. En la figura 29 se observan los dos usuarios descritos.

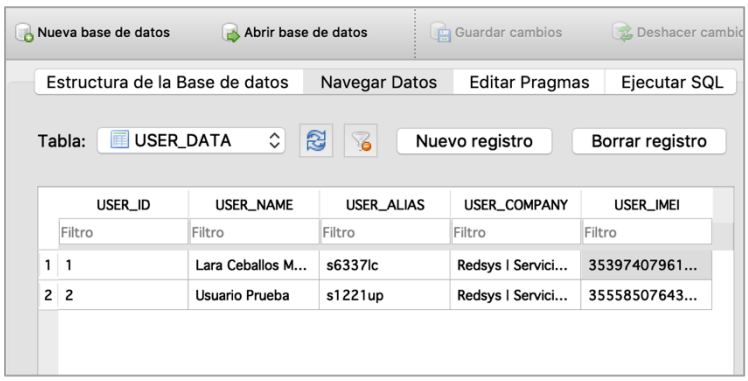

Fig. 29. Tabla de usuarios

Para esta prueba se ha hecho uso de un teléfono móvil externo, cuyo IMEI es 35559507767****. Se puede comprobar en la figura anterior que no se corresponde con ninguno de ellos.

Cuando se instala la aplicación en este teléfono móvil, después de aceptar los permisos que se han comentado en la lista de requisitos funcionales para la aplicación Android, la aplicación accede a la base de datos y detecta que el IMEI no coincide con ninguno de existentes en ella, por lo que lanza un mensaje de advertencia señalando este problema. Cuando el usuario acepte ese mensaje, la aplicación se cerrará. Si el usuario vuelve a iniciar la aplicación, esta volverá a lanzar este mensaje y se cerrará; esto se repetirá tantas veces como el usuario repita la acción.

En la figura 30 de la página siguiente se expone el mensaje explicado.

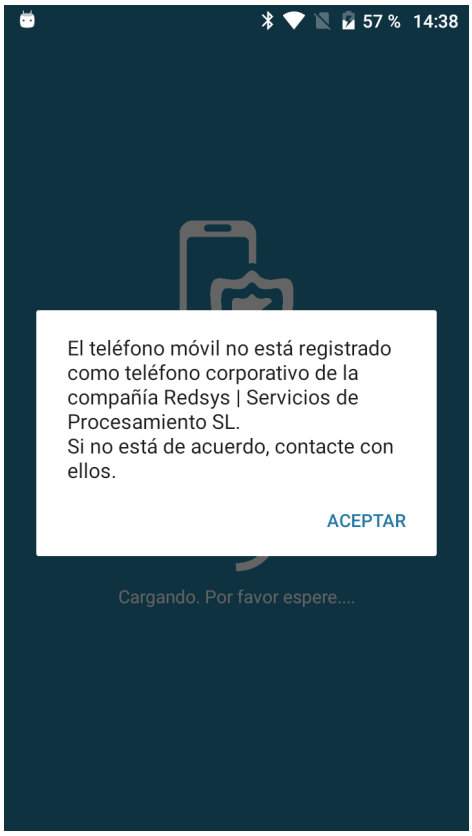

Fig. 30. Mensaje de advertencia, teléfono no corporativo

5.5. Conclusiones del capítulo

En este capítulo se han descrito las pruebas que se han llevado a cabo para garantizar el correcto funcionamiento del sistema de control de acceso, además de su seguridad. Se ha demostrado que el sistema se ejecuta correctamente cuando todos los datos son correctos y, al contrario, se ha comprobado que reconoce cuándo se intenta autenticar un usuario con datos incorrectos.

Por otra parte, se han detectado debilidades en cuanto a la emisión de la señal, ya que es demasiado fuerte. Sin embargo, se ha verificado que es resistente a suplantaciones de identidad evitando que la aplicación se ejecute en teléfonos móviles no registrados.

Por último, se puede concluir que, a partir de la implementación de cada componente del sistema descrita en el capítulo de *Implementación* y las pruebas realizadas en este capítulo, se ha comprobado el cumplimiento de todos los requisitos, tanto funcionales como no funcionales, expuestos en las tablas 3, 4 y 5.

6. ENTORNO SOCIO-ECONÓMICO Y LEGAL

6.1. Impacto socioeconómico

El análisis del entorno social y económico permite obtener una idea aproximada del impacto que conllevaría la aplicación del proyecto en la empresa donde se ha desarrollado, tanto para los empleados como para la compañía en sí.

En primer lugar, desde punto de vista económico, esta solución sería beneficiosa para la compañía ya que, aunque debería comprar nuevos controles de acceso que tuvieran incorporado Bluetooth, supondría la desaparición de las tarjetas de acceso, y por tanto, de la necesidad de mantener el contrato con la empresa que las fabrica. Además, el proyecto propone el uso del teléfono móvil como dispositivo de autenticación frente al control de acceso, lo que otorgaría otra funcionalidad al teléfono móvil que la compañía entrega a todos los empleados y no supondría un gasto adicional.

Otra ventaja para la organización se encuentra en la facilidad de restringir el acceso por horas o días dinámicamente. Esto es útil, por ejemplo, para empleados que tiene una reducción de horas o para el personal que solo trabaje unos determinados días, como pueden ser los trabajadores de otras empresas que viene a colaborar en proyectos específicos. Así se aumentaría la seguridad de toda la empresa, ya que habría un seguimiento individual de los horarios de los trabajadores y se evitarían los accesos en horarios no autorizados.

En cuanto al entorno social, como se ha comentado en la introducción, el mundo vive en la era de la sociedad de la información, la cual es incapaz de separarse de su teléfono móvil. Por tanto, este proyecto supone una gran ventaja para los trabajadores de la empresa, que ya no se tendrán que preocupar de olvidarse el “autenticador” que les permite acceder a ella. Por otra parte, también genera un beneficio para la compañía, puesto que, debido a la necesidad del teléfono móvil para moverse por las instalaciones, obliga a todos los empleados a estar disponibles en todo momento, aunque eso puede suponer una desventaja para ellos.

6.2. Marco regulador

En cuando al marco regulador, se hace un estudio sobre las leyes y reglamentos que se deben tener en cuenta en este proyecto, además de un análisis de los estándares técnicos incluidos.

Debido a que se tratan datos de carácter personal, como es el nombre del empleado y su identificador dentro de la empresa, se ha de cumplir el Reglamento 2016/679[2]. La empresa que haga uso de esta aplicación, en este caso Redsys SL, debe informar a sus empleados de los datos que se tratan y estos deben dar su consentimiento explícito. En caso de que se incumpliera alguna de las normativas expuestas en este Reglamento, las sanciones a la empresa pueden llegar a alcanzar los 20 millones de euros o el 4% de su facturación anual, dependiendo de la gravedad de la infracción.

Por otra parte, la tecnología Bluetooth trabaja en la banda de radio frecuencia ISM, en concreto la banda de 2400-2483,5 MHz. De acuerdo con la nota UN-85 del Cuadro Nacional de Atribución de Frecuencias (CNAF) publicada en la Orden ETU/1033/2017[3], esta banda es de uso común, por lo que no se requiere licencia ni título habilitante para su utilización.

Bluetooth sigue su propio estándar técnico basado en las especificaciones de [5]. Actualmente, estas especificaciones alcanzan la versión 5.0, en la que se incluyen mejoras y nuevas funcionalidades de todos los protocolos desarrollados en la versión 4.0 y que se han mencionado en el capítulo 2.

Por último, cabe mencionar los estándares a partir de los que se han implementado el protocolo de autenticación. Estos son el protocolo EAP [7] y, en concreto, el EAP-PSK [10], que han añadido mayoritariamente interoperabilidad y fiabilidad a la solución, puesto que son protocolos que han sido probados anteriormente y que son compatibles con otras redes inalámbricas como la red WLAN.

[2] Reglamento (UE) 2016/679 del parlamento europeo y del consejo de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos (DOUE-L-2016-80807).

[3] Orden ETU/1033/2017, de 25 de octubre de 2017, por la que se aprueba el cuadro nacional de atribución de frecuencias (BOE-A-2017-12318).

6.3. Planificación

Para planificar el proyecto, se ha realizado un desglose de todas las tareas dividiéndolas por fases con el fin de organizar el proceso y determinar los plazos para la consecución exitosa del proyecto.

La fecha de inicio se fija en el 15 de octubre de 2018 y la de finalización el 19 de febrero de 2019, por lo que la duración del proyecto es de 4 meses y 4 días, 18 semanas. Una vez decidido esto, se clasifican las tareas en las siguientes fases:

- Fase de documentación: se dedica a la búsqueda y lectura de la información útil para el proyecto. Para esta fase se fija un plazo de 2 semanas.
- Fase de planificación y diseño: una vez obtenida la documentación necesaria para la realización del TFG, se procede a definir los procesos que se llevarán a cabo en las siguientes fases y los que quedarán fuera del proyecto. Para esta fase se necesita una semana, aunque estará presente durante todo el proyecto para posibles cambios en el diseño.
- Fase de desarrollo: se dedica a la implementación de la solución diseñada en la fase anterior, se lleva a cabo el desarrollo de la aplicación móvil, el programa software de la placa y el protocolo de autenticación. Debido a la magnitud de esta fase, se fijan 2 meses, es decir, 8 semanas.
- Redacción de la memoria: esta fase se lleva a cabo en 1 mes y medio (7 semanas), y estará mezclada con las últimas pinceladas de la fase anterior.

Por último, se ha realizado un diagrama de Gantt que permite visualizar de manera clara la planificación que se ha expuesto anteriormente.

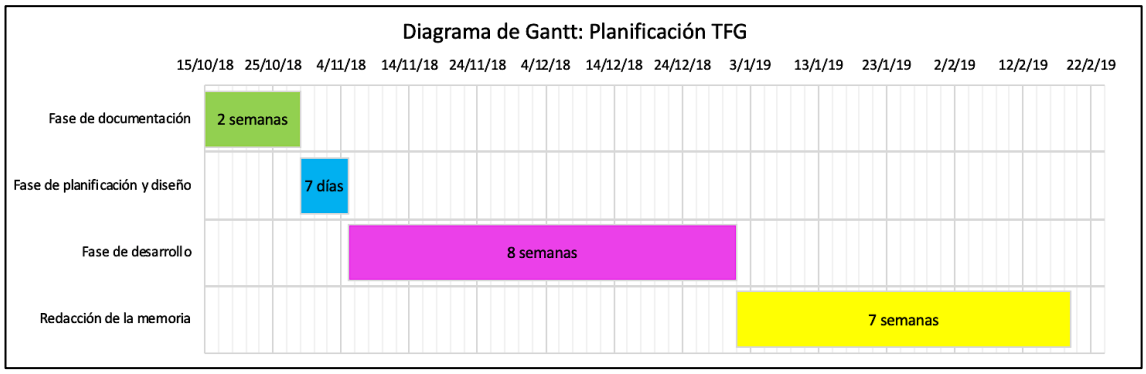

Fig. 31. Diagrama de Gantt

6.4. Presupuesto

Para la elaboración del presupuesto, se han clasificado los costes en dos secciones:

- Costes personales: se corresponden con el salario de un becario en la empresa, Redsys Sistemas de Procesamiento SL, en la que se ha realizado este proyecto, y el coste por el trabajo realizado por los dos directores del proyecto, es decir, las tutorías otorgadas al alumno. Para calcular las horas de trabajo del alumno se han tenido en cuenta las 25 horas semanales que se trabajan como becario más 6 horas semanales adicionales de tutorías y trabajo en casa. Se asume que, desde el comienzo del proyecto hasta el final, todas las semanas ha sido iguales, incluyendo los días festivos como días laborales. Para los directores, se ha calculado un total de 45 horas.
- Costes materiales: se dividen en costes de software y costes de hardware.
  - Software: incluye todos los programas que se han utilizado para el desarrollo software del proyecto, Android Studio, Eclipse y el propio programa de la placa de desarrollo. Sin embargo, todos ellos son gratuitos, por lo que su coste es nulo.
  - Hardware: se corresponden con los gastos de compra del teléfono móvil utilizado para las pruebas, un Samsung Galaxy A3, la placa de desarrollo ESP32 DevKitC V4, un LED y un zumbador que simulan la apertura de la puerta. El teléfono inteligente tiene normalmente una vida útil de 60 meses, por lo que para elaborar el presupuesto, solo se deben tener en cuenta los meses relativos al proyecto.

Por otra parte, se debe añadir un apartado de costes indirectos que suponen un 20% del total presupuestado y en los que se incluyen gastos generales como la conexión a Internet o el precio de la luz.

En la siguiente tabla se muestra el presupuesto total desglosado, donde se observa que el coste total del proyecto es 4.537,59€.

COSTE MATERIAL

Software	Coste total (€)			Coste real (€)
Android Studio	0,00 €			0,00 €
Eclipse	0,00 €			0,00 €
SDK ESP32	0,00 €			0,00 €
Subtotal Software				0,00 €
Hardware		**Vida útil (meses)**	**Amortización**	
Amortización Samsung A3	259,9	60	4	17,33 €
ESP32 DevKitC V4	16,00 €			16,00 €
Tmb12a05 zumbador	1,50 €			1,50 €
LED Tricolor	2,50 €			2,50 €
Resistencia 220 Ohmnios	0,20 €			0,20 €
Subtotal Hardware				37,33 €
TOTAL MATERIAL				37,33 €

COSTE PERSONAL

Concepto	Horas totales	Coste/hora	Coste total (€)
Horas trabajadas en empresa	450	5,50 €	2.475,00 €
Tutorías y trabajo en casa	108	5,50 €	594,00 €
Tutorías directores	45	15,00 €	675,00 €
TOTAL PERSONAL			3.744,00 €

COSTES INDIRECTOS

Concepto	Total costes personal y material	Porcentaje	Coste total (€)
Gastos generales	3.781,33 €	20%	756,27 €
TOTAL INDIRECTOS			756,27 €

TOTAL	4.537,59 €

Fig. 32. Presupuesto desglosado

7. CONCLUSIONES

7.1. Objetivos cumplidos

El objetivo principal de este proyecto era la implementación de un sistema de control de acceso destinado a empresas. A partir de la documentación aportada en los capítulos anteriores, se puede concluir que se ha cumplido con este objetivo.

Además del objetivo principal, cabe destacar algunos de los objetivos secundarios más importantes que se han llevado a cabo con éxito, y son los siguientes:

- Se ha establecido la conexión Bluetooth entre los terminales, en concreto, utilizando Bluetooth Low Energy.
- Se ha conseguido que el proceso de conexión y autenticación sea rápido, y haciendo que la experiencia del usuario no resultara peor que con el actualmente mecanismo de la empresa.
- Se ha incluido un protocolo de autenticación simple, es decir, solo de uno de los dispositivos. El protocolo está basado en las especificaciones EAP-PSK, lo que supone mayor seguridad, ya que es un estándar probado anteriormente.
- La aplicación móvil resulta fácil de utilizar ya que solo se necesita pulsar la notificación de la barra de estado para que se lleve a cabo la autenticación.
- Se han cubierto los principales problemas de seguridad, como son la suplantación de identidad y los ataques Man-in-the-middle.

7.2. Líneas futuras de trabajo

En relación con las líneas futuras por las que se podría seguir con este proyecto, considero que la fundamental es continuar aumentando la seguridad del sistema, debido a que los sistemas de control de acceso constituyen la primera barrera que los atacantes deben superar si quieren acceder a instalaciones o edificios. Las mejoras podrían ser las siguientes:

- Conseguir disminuir la señal Bluetooth de la placa de desarrollo, ya sea a partir de configuraciones internas o recubriendo el sistema físicamente para aislarlo.
- Conseguir implementar el protocolo de autenticación en su totalidad, diversificando la clave compartida y añadiendo robustez al mecanismo que calcula las funciones de resumen.
- Eliminar la descarga de los datos de todos los usuarios en todos los teléfonos. Esto se podría conseguir cambiando la función de descarga por peticiones

REST personalizadas al servidor que gestiona la base de datos, de este modo se limitaría el acceso únicamente a los datos del usuario asociado a cada teléfono.

Por otra parte, en el caso de que el sistema de control se llegara a implementar en la empresa Redsys, se debería modificar la base de datos general de los usuarios activos de la empresa para añadir la clave compartida y el IMEI de los teléfonos corporativos. Además, se debería conectar esta base de datos con la aplicación y el dispositivo de control para que obtengan de ella los datos de los usuarios. En el caso de la aplicación, como se ha comentado anteriormente, solo se obtendrían los datos del usuario asociado al teléfono y en el caso del control de acceso, se accedería a la base de datos en cada autenticación para verificar la clave y el IMEI recibidos son los correctos. De esta forma, se conseguiría realizar una doble verificación de los usuarios, ya que ambos dispositivos estarían actualizados y verificando las identidades de los mensajes de conexión y autentificación que recibiera.

Por último, en cuando a la calidad de la aplicación, se podría seguir mejorándola elaborando estudios sobre el uso de la batería en distintos móviles, para saber si el servicio, que está continuamente funcionando, le afectaría mucho. En caso de que sí lo hiciera, una solución es acotar la activación del servicio a la ubicación de la empresa, por lo que solo arrancaría cuando la aplicación detectara que está cerca de ella.

BIBLIOGRAFÍA

[1] Kisi, «BLE/Bluetooth Technology - How It Works in Access Control,» [En línea]. Available: https://www.getkisi.com/guides/bluetooth-access-control. [Último acceso: 16 Febrero 2019].

[2] Nepdap, «Mace,» [En línea]. Available: https://www.nedapidentification.com/es/productos/mace/. [Último acceso: 16 Febrero 2019].

[3] ESPRESSIF SYSTEMS (SHANGHAI) CO., «ESP32-WROOM-32 Datasheet,» 2018.

[4] Bluetooth SIG, Inc., «Topology Options,» [En línea]. Available: https://www.bluetooth.com/bluetooth-technology/topology-options. [Último acceso: 8 Enero 2019].

[5] Bluetooth SIG, Inc., «Bluetooth Core Specification v5.0,» 2016.

[6] Bluetooth SIG, Inc., «GATT Specifications,» [En línea]. Available: https://www.bluetooth.com/specifications/gatt. [Último acceso: 28 Enero 2019].

[7] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, IETF RFC 3748, «Extensible Authentication Protocol (EAP),» Junio 2004. [En línea]. Available: https://tools.ietf.org/html/rfc3748. [Último acceso: 24 Noviembre 2018].

[8] Cisco, «Extensible Athentication Protocols,» 18 Febrero 2018. [En línea]. Available: https://www.cisco.com/c/en/us/td/docs/net_mgmt/prime/access_registrar/6-1/user/guide/user_guide.html. [Último acceso: 28 Enero 2019].

[9] M. Nystroem, IETF RFC 4793, «The EAP Protected One-Time Password Protocol (EAP-POTP),» Febrero 2007. [En línea]. Available: https://tools.ietf.org/html/rfc4793. [Último acceso: 24 Noviembre 2018].

[10] F. Bersani, H. Tschofenig, IETF RFC 4764, «The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method,» Enero 2007. [En línea]. Available: https://tools.ietf.org/html/rfc4764. [Último acceso: 24 Noviembre 2018].

[11] Android Developers, «Introducción a Android,» 31 Enero 2019. [En línea]. Available: https://developer.android.com/guide/. [Último acceso: 4 Febrero 2019].

[12] Android Developers, «Android Studio,» [En línea]. Available: https://developer.android.com/studio/?hl=es-419. [Último acceso: 4 Febrero 2019].

[13] Android Developers, «Aspectos fundamentales de la aplicación,» 31 Enero 2019. [En línea]. Available: https://developer.android.com/guide/components/fundamentals. [Último acceso: 4 Febrero 2019].

[14] S. Jasek y SecuRing, «GATTacking Bluetooth Smart Devices,» [En línea]. Available: http://gattack.io/whitepaper.pdf. [Último acceso: 16 Enero 2019].

[15] Android Inc., «Opciones de almacenamiento,» 17 Enero 2019. [En línea]. Available: https://developer.android.com/guide/topics/data/data-storage#filesInternal. [Último acceso: 29 Enero 2019].

[16] Google Cloud, «Generating a Strong Pre-shared Key,» 12 Septiembre 2018. [En línea]. Available: https://cloud.google.com/vpn/docs/how-to/generating-pre-shared-key. [Último acceso: 25 Noviembre 2018].

[17] ESPRESSIF SYSTEMS (SHANGHAI) CO., «ESP32 Modules and Boards,» [En línea]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/hw-reference/modules-and-boards.html#esp32-modules-and-boards. [Último acceso: 4 Enero 2019].

[18] Bluetooth SIG Inc., «Alert Notification Service,» 15 Septiembre 2011. [En línea]. Available: https://www.bluetooth.com/specifications/gatt. [Último acceso: 28 Enero 2019].

ANEXO A: Tabla de tiempos

Prueba	Autenticación	Tiempo (seg)
1	Correcta	2,25
2	Correcta	2,05
3	Correcta	2,38
4	Correcta	2,15
5	Correcta	2,30
6	Correcta	1,95
7	Correcta	2,42
8	Correcta	2,25
9	Correcta	2,31
10	Correcta	2,31
11	Correcta	2,26
12	Correcta	2,23
13	Correcta	2,08
14	Correcta	2,25
15	Correcta	2,00
16	Correcta	1,90
17	Correcta	2,25
18	Correcta	2,35
19	Correcta	1,90
20	Correcta	2,22
21	Correcta	2,31
22	Correcta	2,25
23	Correcta	2,21
24	Correcta	2,23
25	Correcta	2,20

ANEXO B: Project summary

1. Introduction

In this work an authentication system has been developed for physical access control based on a Bluetooth connection between a microprocessor, which will be a prototype of the access control, and a smartphone with Android operating system. In addition, the motivation of creating this authentication system arises from the need to update the access control technology of the company that has collaborated in this TFG, Redsys.

2. Goals

This main objective explained in the last section is divided into several sub-objectives that help to better understand the project that has been developed:

- Development of a prototype of access control using a microprocessor which authenticates devices through Bluetooth technology.
- Design and implementation of a mobile application for Android that intervenes in the authentication process and plays the role of "authenticator" against access control.
- Definition and implementation of the secure communications protocol that will be used in the connection between access control and mobile phone.

3. State of Art

3.1. Access control

Particularizing the subject that is exposed in this work, an access control is a service that regulates the entrance of people to buildings, plants and / or departments of a company.

Within the controls using Bluetooth connection, the authentication system is composed of a smartphone and a control device. The phone has a mobile application installed that allows establishing a connection with the access control device, and the authentication is performed through the exchange of a shared secret.

On the other hand, the development board called ESP32 DevKitC V4 has been chosen to develop the prototype of the access control device. This microprocessor has been chosen due to the following characteristics:

- Integration of Bluetooth, both Classic and Low Energy, in the module, so it can work as a complete independent system.

- Includes an SDK (Software Development Kit) with specific source code for the board and examples of how to use it. In addition, it works in C language, which facilitates the development of applications.
- Possibility of implementing cryptography, includes a security section with flash encryption and hardware cryptographic functions (SHA, AES, ECC).

3.2. Bluetooth

Bluetooth is a short-range wireless communications protocol created in 1994 by Jaap Haartesen and Mattisson Sven, and published through Bluetooth Special Interest Group (BSIG), for the purpose of being used in low power devices. It works in the radio frequency band ISM (Industrial, Scientific and Medical), between the bands 2402-2480 MHz and 2400-2483.5MHz.

On the other hand, Bluetooth has several specifications that modernize the original standard. Currently, all devices, both mobile phones and access controls, have integrated versions from 4.0. In this version, the protocols *Classic Bluetooth*, *Bluetooth High Speed* and *Bluetooth Low Energy (BLE)* are included.

The last one provides the same range of coverage as the previous ones, but considerably reducing battery consumption. Access controls require quick communication since many people accumulate at the doors at the start and the end time of workdays. Therefore, these systems use BLE technology, which has a connection time much lower than traditional Bluetooth and improves the exchange of messages due to the division of these into small packages.

3.2.1. Bluetooth Low Energy

As in *Classic Bluetooth*, the protocol stack has two main parts: the Controller and the Host. At the same time, the Controller is divided into the physical layer (PHY) and the link layer (LL), and the Host into the following protocols and profiles:

- L2CAP (Logical Link Control Protocol): supports the multiplexing, segmentation and reassembly of packets in higher level protocols.
- GAP (Generic Access Profile): defines the generic procedures for the discovery of devices and the connection between them, as well as a common parameter format for the user interface.

- ATT (Attribute Protocol): defines the protocol for the discovery, reading and writing of attributes in a device.
- GATT (Generic Attributes): describes a hierarchical data structure built on the ATT protocol.
- SM (Security Manager): defines the protocol and behavior of the pairing, authentication and encryption between devices.

GAP, ATT and GATT will have critical functions in the exchange of messages between the devices of the access control system.

3.3. Authentication protocol

The authentication process is carried out by sending a challenge from one of the devices, the "authenticator", to the other, which is the one that wants to be authenticated, and depending on the response of this, the authentication will be correct or not. This process can be done in many ways, including in a manner more or less complex. In this work, the standard defined in RFC 3748 called Extensible Authentication Protocol (EAP) will be taken as a reference, due to the fact that it includes a wide variety of authentication methods. Two of the most suitable methods for access control are the following ones:

- EAP Protected One-Time Password (EAP-POTP): it consists of OTP tokens (One-Time Password) which are introduced in the EAP authentication process. These tokens are generated from a shared key or seed, which is known at both ends. The protocol provides client authentication to the server in simple mode, but it can also provide mutual authentication if necessary.
- EAP Pre-Shared Key (EAP-PSK): provides a secure communication channel thanks to the mutual authentication of the ends. It is based on the derivation of static and session keys from another key that both devices know before starting the protocol. This avoids having to negotiate the cryptographic parameters to be used during the exchange of messages.

3.4. Android Application

An Android application is composed of components which are individual entities that act independently between them and fulfill a specific role within the applications. There are 4 different types of components:

- Activities: they represent the interfaces on which the user can interact with the application.
- Services: they refer to the components that run in the background and perform long tasks. Therefore, they do not provide user interfaces.
- Content providers: they manage the shared application data sets, the information which Android system provides about the phone or the user and the private application data.
- Broadcast receivers: they are the administrators who respond to message announcements throughout the system.

4. Design of the solution

The sequence of activities followed by the access control system is:

1. The phone requests access to the database where the employee's data and the general data of the application are located.
2. A copy of the necessary data of the database is made on the phone.
3. The development board is continuously being announced. Once it is identified, the phone sends the connection request.
4. The development board accepts the connection and starts the EAP-PSK authentication protocol, sending the corresponding data to the first message of the protocol.
5. The mobile phone sends the second message of the protocol, which includes the MAC which authenticates the user.
6. The development board processes the MAC and depending on the result, it will open the door or not. This is represented by the lighting LED and the buzzer.

4.1. Access control design

As mentioned above, for this project an ESP32 DevKitC V4 development board has been chosen as the central access control device. In addition, to signal the correct or incorrect authentication of the user in front of the access control, a tricolor LED RGB and a buzzer are being used. When the authentication is correct, the LED lights green and the buzzer beeps for 500 milliseconds, while if it is incorrect, the LED lights red and the buzzer emits a different sound during the same period of time.

Regarding the software implemented inside the development board, it is mostly based on the Bluetooth Low Energy specifications. Therefore, it must have a GATT profile, which

contains at least one service with two characteristics. The first feature must have the "read-only" property and will contain the data of the first authentication protocol message for the mobile application to read. The second feature must have the "write-only" property and will receive the data from the second message of the protocol sent by the mobile application.

4.2. Android Application

The Android application that has been created for this project is called AMB (Access Mobile Bluetooth). It has the function of providing the necessary information about the user who owns the application so that the control identifies him/her as a company worker and accepts his entry. The different activities that are developed in the application are described as follows:

- Introduction of credentials: there is a table of the database in which each corporate mobile phone is associated to a single employee of the company thanks to the IMEI (International Mobile Equipment Identity) of the phone. When the application recognizes this value, it identifies the user of the mobile without the need to register.
- Search for devices and connection: the application is implemented as a service that is constantly scanning for the access control device. Once it finds this device, it launches a notification on the screen indicating if the user wants to connect. By clicking on the notification, the connection is accepted, and the user's authentication process begins. Although it is not necessary that the application is opened for the service to be working, in case the user had it on, he/she can connect with the control device by clicking a button inside the application.
- Activity records: each time the user connects to an access control, the application saves the process and displays it on the tab "Register Access" flyout.
- System preferences: a settings tab is included in which some aspects of the application can be modified.
- Closure of the application: there are two options to close the application. In the lateral tab there is the option of "closing the application" that is used to exit it and end the service. The second option is to close the application

through the back button, which means that the device scanning service continues to work in the background.

An important point of the application is the database, which consists of 3 tables containing:

- Users data: name, identifier within the company, company in which the person works, IMEI of its corporate telephone and key shared with the access control.
- Control devices data: name, MAC address and alias, which will be used to show in the access record.
- Access record data: alias of the access control device, date and time of the access, result of the connection and indicator of entry or exit.

When the application starts for the first time, it downloads its configuration and makes a copy of the database structure with empty tables. Then each component of the application accesses the original database when they need it and copies the appropriate data in the empty tables. Once this is done, whenever that data is required, the database of the telephone is accessed, which prevents continuous access to the database located in the cloud.

4.3. Authentication protocol

The authentication protocol on which the implementation of this project is based is the EAP-PSK, explained in the *State of Art* section. However, instead of including mutual authentication, it has been simplified to a simple user authentication against the access control to improve the user experience. This is achieved due to the fact that a lower number of messages are exchanged, thus the door opening speed increases.

The exchange of messages can be summarized in this figure:

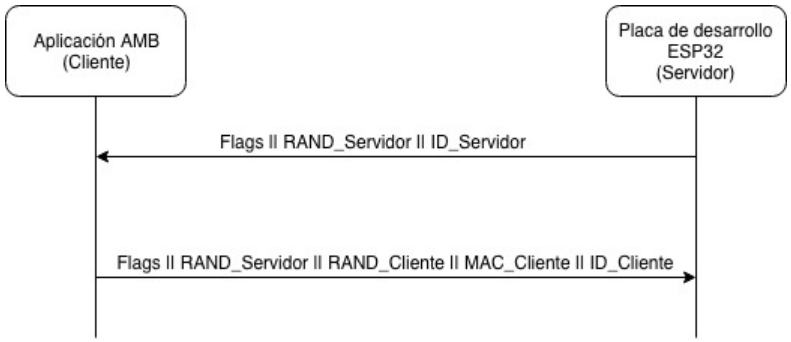

Fig. 33. Authentication protocol scheme

4.4. Security Analysis

In the latest Bluetooth updates, mechanisms to provide security in the connection between devices have been implemented. Despite having 3 safe modes, it has been proven that they are not reliable. Because of this, it has been decided to use an alternative authentication protocol implemented over the Just Works mode of Bluetooth. As mentioned before, the protocol chosen is the EAP-PSK because it provides the following security services: user authentication, integrity, protection against replay attacks and dictionary attacks.

Regarding the mobile application, the biggest security problem that exists in a mobile phone is the theft of confidential data. In order to avoid this, the development of access control application, internal storage and the private SQLite database have been used for the storage of user data, authentication keys and registry data stored by the application, thus it is not possible to access this information from other applications, or from the mobile phone itself.

To avoid identity theft, as indicated above, the IMEI of the mobile phone has been associated with the user, so that only phones delivered by the company are suitable for installing the application and proceeding with the authentication. In addition, once the application is working, the user must explicitly accept the connection in order to authenticate against the access control, which avoids the MITM attack.

6. Tests performed

This section includes all the different tests that have been done throughout this project, checking the correct behavior of the entire access control system.

The first test performs a correct authentication between the mobile application and the development board. When the authentication finishes, the LED lights in green and the correct exchange of messages can be checked in the monitor of the development board. Moreover, this test is also made to check the duration of the connection and authentication processes, which finishes sequentially after 2,2 seconds. Therefore, it can be concluded that the duration is equal or less than with the mechanism currently used in the company, so the user experience is better.

The second test performs an incorrect authentication between the devices by introducing an error in the reading of the server random number. This implies that the MAC is calculated incorrectly and, therefore, does not coincide with the one calculated later by

the development board. This test ensures that the program implemented on the development board is resistant to sending incorrect data by the mobile application.

The third test tries to check that the area covered by the Bluetooth signal on the development board is as small as possible. There is a function in the Bluetooth libraries of the development board that allows to modify the power of the signal. However, this modification is not enough for the system, due to the test shows that the range of the Bluetooth communication even at minimum power still reaches 10m, which results in a security problem.

The last test checks that non corporative phones cannot start the application and therefore, users who do not belong to it cannot violate the security of access control. The result of this test is successful because a warning message comes up in the external phone when it tries to activate the application.

7. Conclusion

To complete this summary, it can be concluded, based on the documentation provided in the previous chapters, that the principal objective has been met. Moreover, it is worth highlighting some of the most important secondary objectives that have been carried out successfully, and they are the following:

- The Bluetooth Low Energy connection between the terminals has been established.
- A simple authentication protocol has been included.
- The mobile application is easy to use since you only need to press the notification of the status bar for authentication to take place.
- The main security problems have been covered, such as identity theft and Man-in-the-middle attacks.

Regarding future work, I believe that the key is to continue increasing the security of the system because access control systems are the first barrier that attackers must overcome if they want to access facilities or buildings. Moreover, in case the control system was to be implemented in the Redsys company, the general database of the active users of the company should be modified to add the shared key and the IMEI of the corporate telephones. In addition, this database should be connected to the application and the control device so that they can obtain user data from it. In the case of the application, as mentioned above, only the user's data associated with the

telephone would be obtained and in the case of access control, the database would be accessed on each authentication to verify the key and the IMEI received are the correct ones.